



Principles of Design Diagramming

Vasily Pantyukhin

Advisory Solutions Architect

EMC EMEA Global Professional

Services Delivery

Vasily.pantyukhin@emc.com

EMC²

Table of Contents

1	Introduction.....	3
2	What is Design Diagramming?	4
	Diagramming is brain-friendly encoding	5
	Categories of diagramming.....	6
	General types of diagrams.....	7
	Components of diagrams.....	9
	Section summary.....	11
3	What general principles make diagramming efficient?	12
	Define the goal	12
	Define the context.....	13
	Choose appropriate encodings	20
	Explain encoding	21
	Be consistent in patterns.....	23
	Use unity methods	24
	Use variety methods	26
	Make progress iteratively	27
	Section summary.....	28
4	How to make diagrams better?.....	29
	Use different visual channels	29
	Make diagrams compatible with the reality.....	29
	Define structure	31
	Create symmetrical visual patterns and structures.....	34
	Use whitespace	35
	Create complex based on simple.....	36
	Decrease noise.....	39
	Make diagrams stylish and esthetically attractive.....	40
	Use colors right.....	41
	Use the right fonts.....	44
	Use paper, whiteboard, and everything you want.....	44
	Section summary.....	45
5	Conclusion	46
6	Bibliography.....	47

Disclaimer: The views, processes or methodologies published in this article are those of the author. They do not necessarily reflect EMC Corporation's views, processes or methodologies.



Heliocentric Universe diagram; Nicolaus Copernicus, 1543

1 Introduction

Modern enterprise IT infrastructure consists of an enormous number of objects described by tremendous amount of characteristics. Connections and interdependencies between them are complicated.

Therefore, designers of IT solutions must consider and handle a massive amount of information.

That information has to be analyzed, and results should be presented to the stakeholders. Usually, these are quite heavy tasks. Of course, to increase quality and speed of data investigation and representation, designers use different tools. The diagram is one of the most efficient and proven tools.

As is true of every instrument, diagrams have to be used smart. Unfortunately, the potential of this incredibly helpful tool often is not used to its fullest. Even more, poor diagrams confuse the designer and customers or require too much effort to understand. Consequently, benefits can be significantly reduced.

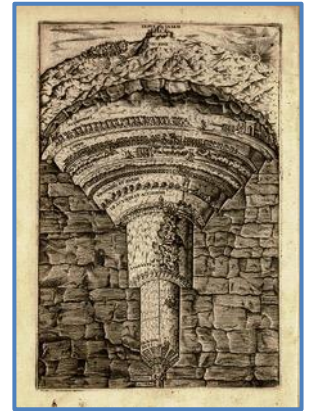
This article focuses on the methods to accelerate and improve the process of diagram development. I will establish the fundamental principles of efficient diagramming. Among these principles, I will:

- describe the difference between exploratory and explanatory diagramming
- discuss different information presentation options
- consider methods of objects' unity and variations

Readers will gain information about how to professionally use diagrams in the context of various tasks.

The principles, techniques, and tricks described in the article are proven by the years of actual project design. Since the v1.0 release, I have mainly focused on planning, implementing, and troubleshooting ViPR® in the EMEA region. During this period, I developed and used dozens of diagrams. That is why examples related to that powerful but complex product illustrate the article.

The article is most relevant to IT architects, pre-sales consultants, and other professionals involved in solutions development and presentation.



Dante's Inferno diagram; Dante Alighieri, 1308-1321

2 What is Design Diagramming?

Famous scientist Herbert Simon said, "Solving a problem simply means **representing** it so as to make the solution transparent." Therefore, the primary task of solution designers is to present design objects, ideas, actions, their connections, and dependencies. According to vocabulary.com, a **diagram is a drawing that shows the different parts of something and how they work together**. This definition supports the notion that a diagram is perhaps the most powerful tool to achieving a designer's task.

This article is devoted to diagrams. However, I do not consider them superior to other forms of communication, just different. We use *charts* and *graphs* to present quantitative information visually. *Tables* are extremely powerful when we need to process a significant amount of data. *Text* is very flexible. However, in solutions design, diagrams can be more useful than other forms in respect to promoting holistic investigation and understanding.

- A diagram is the quickest way to perceive qualitative data.
- A diagram offers both functional and emotional aspects of our knowledge.
- Massive amounts of information can be represented more efficiently. A diagram eliminates unnecessary complexity to highlight the core idea.
- A diagram can help us see patterns in data that may be otherwise hidden or appear disordered.
- A diagram can produce a less vague description than a text, because it forces one to be more structured. By necessity, the notations of the diagramming language, which serve as its vocabulary, restrict what is and is not allowed in the representation. (Nakatsu, 2009)

Each person has their own experience. That background knowledge is the basis for the first interpretation of the diagram. A picture alone is usually clear to people who already know all aspects of the subject. However, designers and stakeholders unfamiliar with the topic and methods of design can misinterpret the picture and make wrong decisions. To avoid that, textual titles, labels, and legends are essential for design diagrams. They must define the rules associated with the representation and explain how to interpret the diagram. So, **visualization is not equal to diagramming**. However, it remains the most significant component.

Diagramming is brain-friendly encoding

The powerful human cognitive system is adapted to survive. Life demands instantaneous reaction to external events. We always need to find something delicious or see a potential danger. That is why evolution developed highly efficient visual pattern identification. This strength of our perception should be used to establish diagramming on identifiable **patterns**. Such an approach enables us to quickly deliver required information with minimum brain effort to recognize and understand.

Diagramming helps overcome the limitations of our brains. Human **short-term working memory** has low capacity and visualization allows us to extend it by caching data with a fast visual channel to access. Only quite simple objects can be stored in short-term memory, so information has to be encoded beforehand in the form of basic shapes.

Modern scientists estimate capacity of our **long-term memory** as several hundred megabytes. We do not remember everything, only a small part of the information that is encoded in about first 24 hours after an event occurs. By its nature, diagrams selectively encode information using common simple objects. That is a cognitive-friendly way for an efficient long-term memory exploitation, better knowledge transfer, and memorization.

As simplified model, we can consider that every person has a limited amount of a brainpower dedicated to the particular subject or problem. Part of this power is used to decode incoming information. The remaining brainpower may be used to understand the message. The task of every solution designer is to **minimize the amount of unnecessary searching and decoding** to convey the message while maintaining as much power as possible to understand and process information.

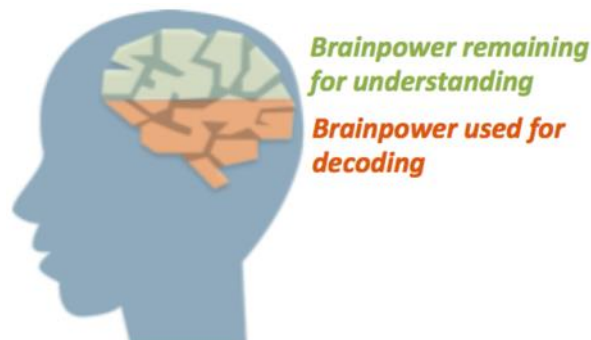


Figure 1: Readers have a finite amount of brainpower to apply to a diagram. Any resource used on decoding reduces what is available for understanding. Adapted from (Illiinsky & Steele, 2011)

Human operates on both concrete and abstract ideas. Abstract concepts cannot be represented as objects in the real world. To simplify sharing and perceiving conceptual thoughts, designers encode them. That helps to **make visible in the mind that which is not visible to the eye**.

Categories of diagramming

There are two categories of data diagramming: eXploratory and eXplanatory.

- **eXploratory** diagramming is appropriate when we need to find what is inside the set of data, reveal something unknown, or find the combination of a solution's components that best satisfy the design requirements. We should exploit human **cognitive abilities** for efficient eXploration.
- **eXplanatory** diagramming is applicable when data has been explored and the idea to share has already been defined. The primary task here is to represent findings to an audience in a way that requires minimal effort and time to understand. In eXplanatory diagrams, we have to take into account human **sensory capabilities**.

In a solutions design, both categories of diagramming are used. Even more, many widely exploited design diagrams are mixture of eXplanatory and eXploratory approaches. On one hand, they should be convenient for eXploration of available information, hypothesis proposal, and considering design choices. On the other, results should not be lost in a mass of unnecessary details. Findings and design decisions need to be clear to other designers and project stakeholders. Design goals and audience preferences that define the **proportion of eXploration and eXplanation** is such a hybrid category.



Figure 2: Proportion of eXploration and eXplanation in diagrams can vary widely.

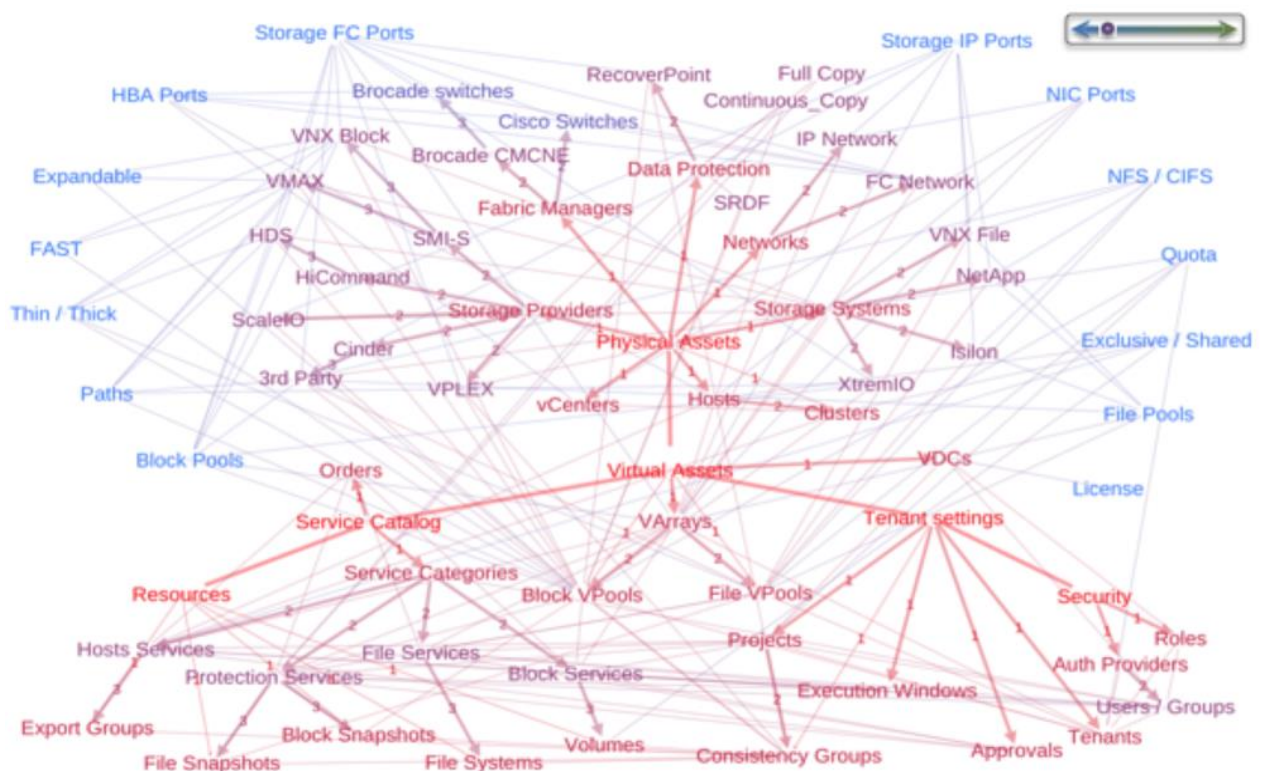


Figure 3: Highly eXploratory diagram; All ViPR components and dependencies presented in the diagram. Reader should discover the needed information by himself.

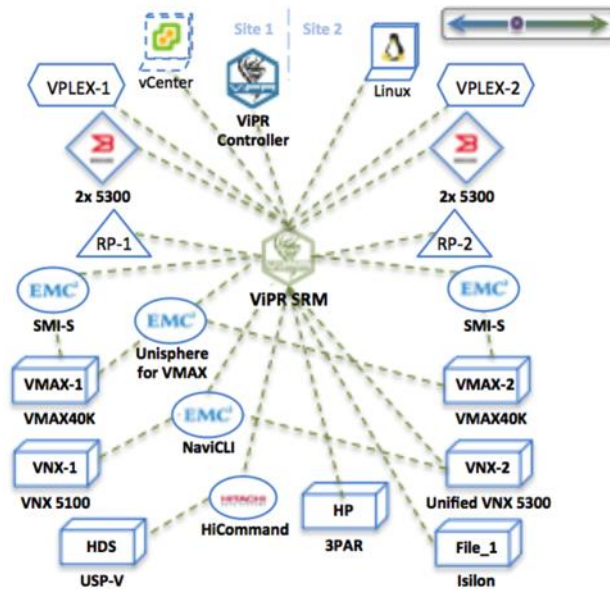


Figure 4: eXploratory diagram; ViPR SRM communicates with components of the solution directly or via different Storage Providers. Reader can investigate particular relations if needed.

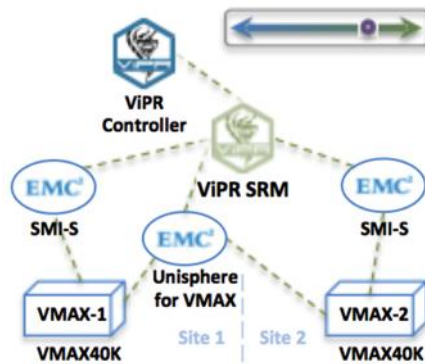


Figure 5: eXplanatory diagram; The central idea is revealed: ViPR SRM communicates with VMAXs via two SMI-S providers and Unisphere®.

General types of diagrams

There is a wide variety diagram types. In design diagramming, three general types are used; Topology, Hierarchy, and Flow.

Topology is defined by (Nakatsu, 2009) as the organization of components in the system. It consists of components, as well as the connections among them.

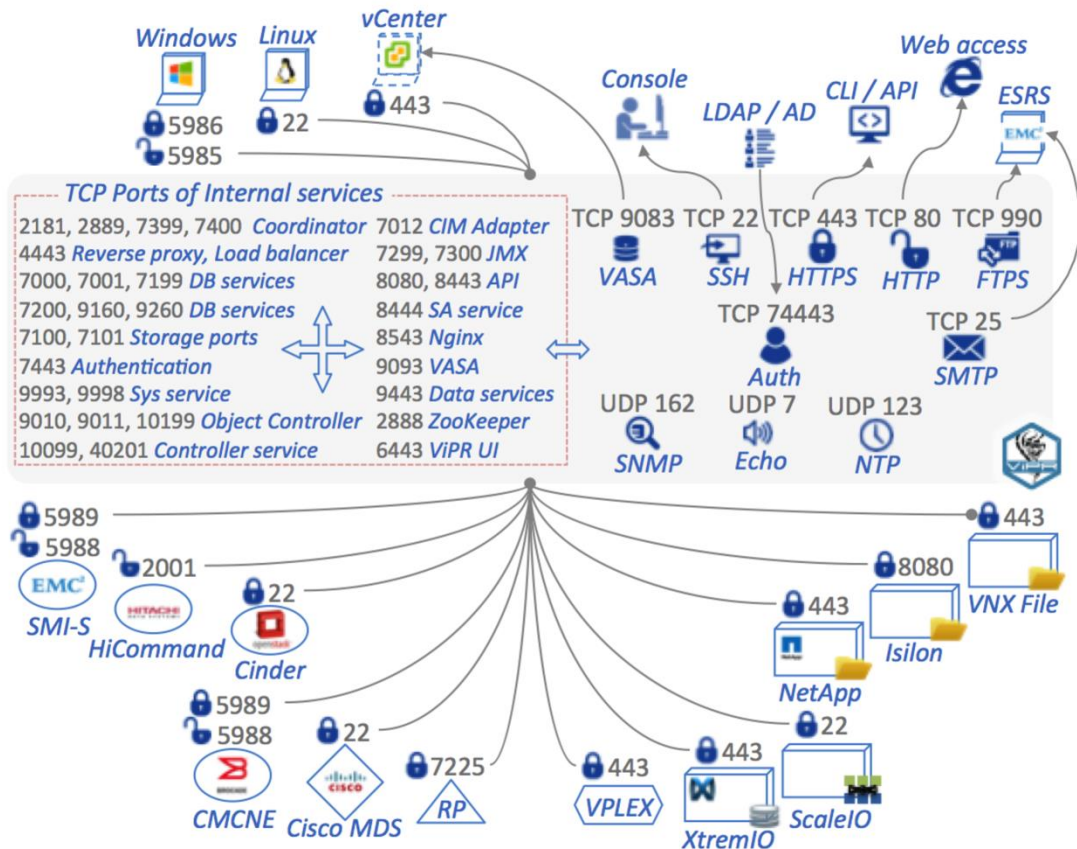


Figure 6: Topology diagram; Components of the diagram are internal and external network ports used by ViPR. Connections organize them into the single structure.

A **hierarchy** shows how different components of the system are subordinated. Note that the topology is different from the hierarchy. Topology describes how same grade (siblings) components are interconnected. Hierarchy always represents relations of different grades (parents-children).

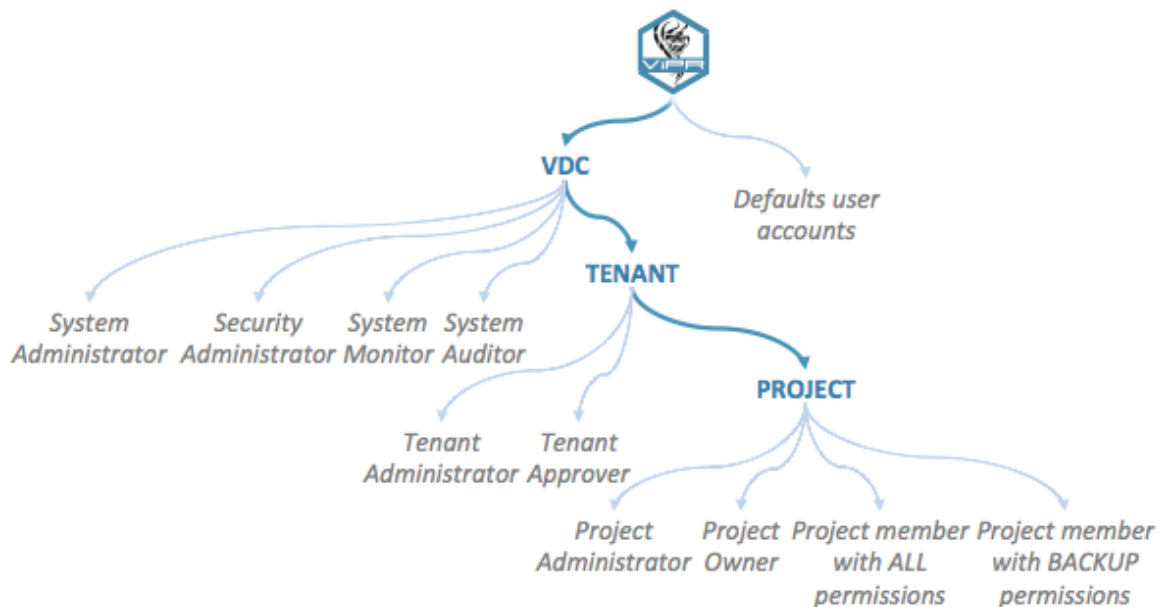


Figure 7: Hierarchy diagram; ViPR Roles and Users represent the structure of subordinated instances.

Flow is a chronological ordering of steps in the process or sequence of actions. The beginning and end of the flow have to be easily identified. Conditional branches and cycles are acceptable.

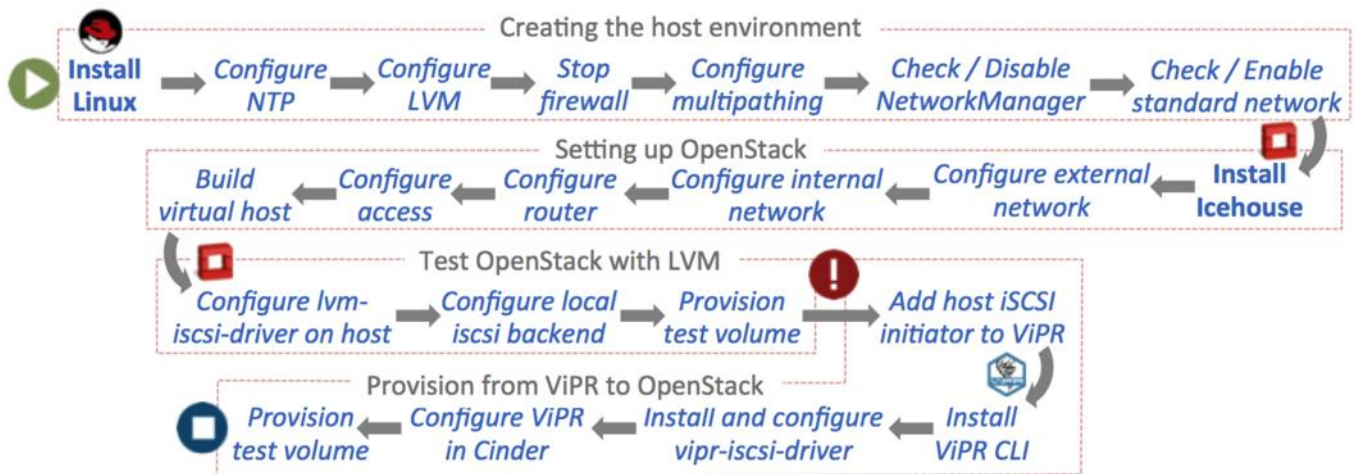


Figure 8: Flow diagram; Steps to configure ViPR as a Storage Provider for an OpenStack are chronologically dependent. They represent the flow of the defined procedure.

Components of diagrams

We can divide all components of diagrams into entities and relations. (Ware, 2013) **Entities** are our design objects. In some cases, a group of things can be considered as a single entity.

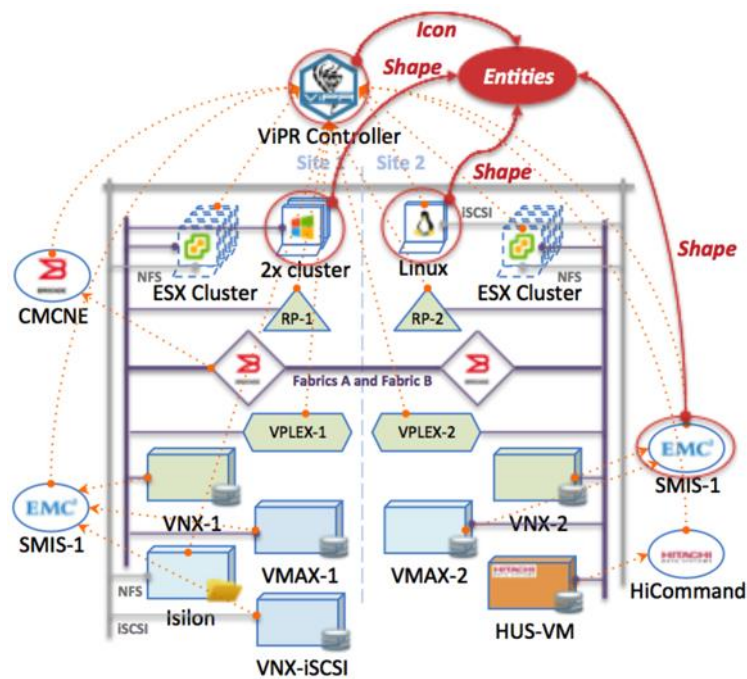


Figure 9: Entities are the objects of our design; in ViPR, solutions design shapes and icons are widely used.

Relations describe how entities are associated with one another. They can be physical or conceptual.

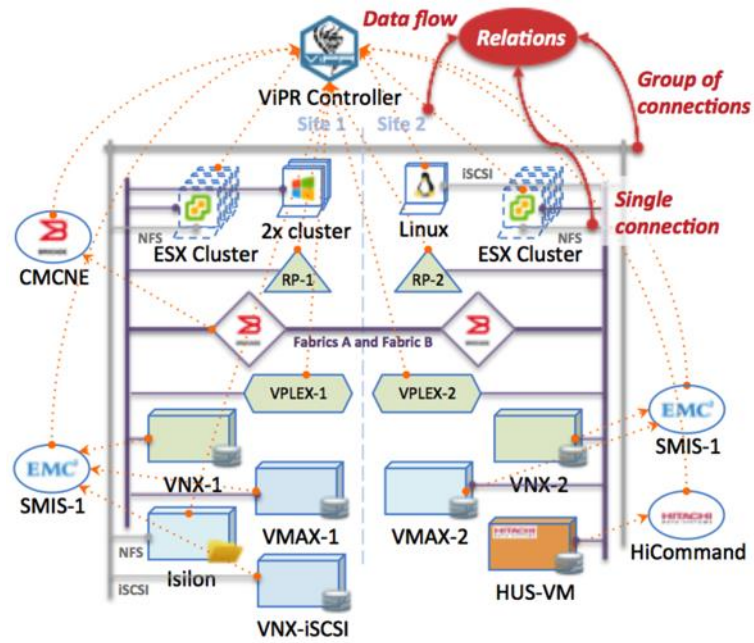


Figure 10: Relations are associations between design objects; Arrows and lines are used in ViPR solutions design to represent components' relations.

Both entities and relations can have **attributes**. Consider an attribute as a property that cannot be used independently. Defining what should be an entity and what should be an attribute is not always straightforward. For example, the IP address can be thought of as an attribute of the host. An IP address can also be thought of as an entity itself when showing the relation between the host and its IP address.

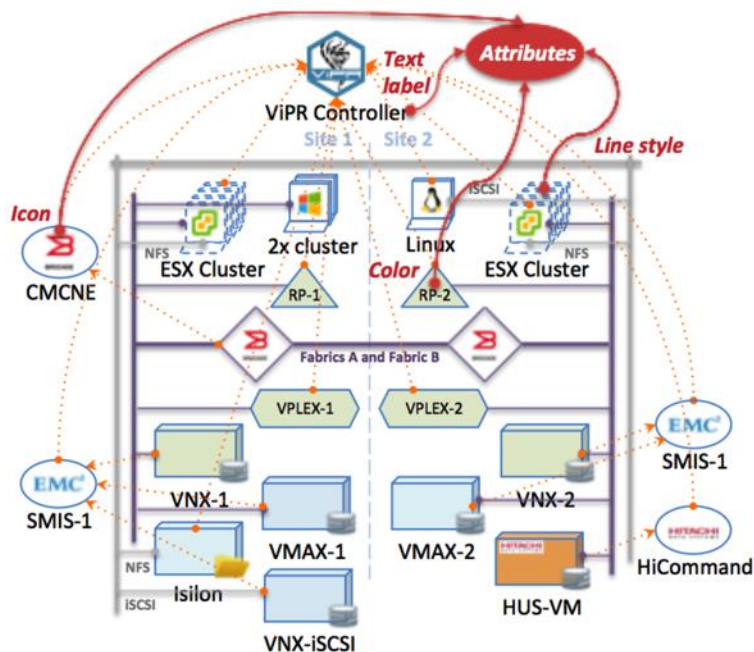


Figure 11: Attributes are properties of design objects and their relations; The many different ways to represent attributes used in ViPR diagrams are explained in details below in the article.

There are three types of attributes:

- **Ordinal** – qualitative data; can be arranged in some order or hierarchy. Examples are: priority – high / medium / low; current state – excellent / good / average / below average / poor / worst.
- **Nominal** – qualitative data; can be characterized by different categories and cannot be ranked or ordered. Examples are: write pattern – sequential / random; RAID level 0 / 10 / 5 / 6
- **Quantitative** – consist of numerical values representing counts or measurements. Examples are: a number of drives in a Disk Array Enclosure – 15, 30, 60; capacity of the volume – 1.9TB.

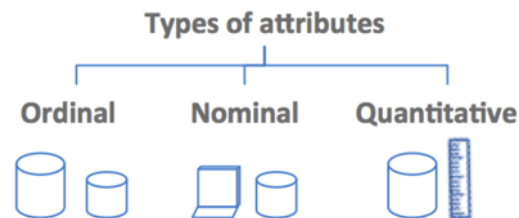
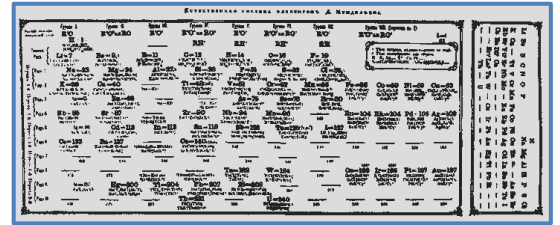


Figure 12: Three types of attributes.

Section summary

- The fundamental goal of diagramming is to help us perform cognitive design work faster and better, or let us explore something new.
- There are two categories of data diagramming: eXploratory, when revealing something unknown, and eXplanatory, when representing the idea to an audience.
- In design diagramming, three general types are used; Topology, Hierarchy, and Flow.
- Entities, Relations, and their Attributes are essential components of diagrams.



Periodic table; Dmitri Mendeleev, 1869

3 What general principles make diagramming efficient?

Define the goal

Diagramming is the tool that helps a designer achieve the primary purpose of the whole design. Every diagram has to have its specific goal. Understanding and stating that **goal is the key to the diagramming success**. Well-defined objectives direct subsequent diagram implementation decisions and help to evaluate the final result.

Ideally, the goal must be defined before diagramming starts and should be stated it in terms of the knowledge that can be obtained from the diagram. Of course, **the particular implementation should not be mentioned**. Selecting in advance one particular way to implement the diagram constrains thinking to within the limits of that decision and can obscure plenty of other possibilities.

To begin defining the purpose of the diagram, ask yourself about the nature of information you are going to work with. According to (Iliinsky & Steele, 2011) related questions should include:

- What values or data dimensions are relevant in this context?
- Which of these dimensions matter; matter most; and matter least?
- What are the key relationships those need to communicate?
- What properties or values would make some individual data points more attractive than the rest?
- What actions might be taken once this information need is satisfied? What values justify that action?

A good statement always has to include an **action verb** followed by the specific **object** of action. Optionally it may contain a **contextual clarifier** to clarify the circumstances or conditions. (Bettencourt, 2010)

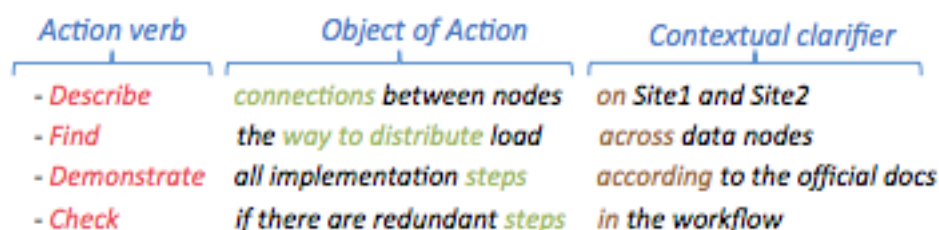


Figure 13: Recommended goal statement structure; Examples of real statements are presented.

The next table is the list of commonly used action verbs that can describe a goal statement for eXploratory and eXplanatory diagrams.

eXploratory	<i>Check</i>	<i>Detect</i>	<i>Determine</i>	<i>Discover</i>	<i>Ensure</i>	<i>Find</i>
	<i>Identify</i>	<i>Learn</i>	<i>Obtain</i>	<i>Plan</i>	<i>Try</i>	<i>Verify</i>
eXplanatory	<i>Advice</i>	<i>Confirm</i>	<i>Demonstrate</i>	<i>Describe</i>	<i>Highlight</i>	<i>Provide</i>
	<i>Remember</i>	<i>Recall</i>	<i>Share</i>	<i>Show</i>	<i>Teach</i>	<i>Understand</i>

Table 1: The list of the verbs usually used for a goal statement.

A contextual clarifier should answer questions that begin with “What”, “Who”, “Why”, “When”, “Where, or “How Much”. Of course, the question “How?” is prohibited.

A Goal statement has to be defined by **unambiguous language** so everyone hearing or reading it, would clearly understand it. Avoid imprecise words such as *usable* or *efficient*.

Define the context

Diagramming should be adaptive to the design process and its context. Sometimes, the design is primitive and simply follows the set of rules and procedures. There are situations when a solution is complex but the general canvas is clear and design is focused on details. In other cases, even the final goals are not defined, and designers make their job in an environment of uncertainty. That is the usual price for freedom to innovate. Each of us has done the work under stressful conditions at least once or in extremely strict time constraints. Described contexts require different approaches in design as a whole and diagramming in particular.

Cynefin (pronounced cun-ev-in) is the framework that helps to identify the context of the design project and advise suitable strategies for dealing with it. Cynefin consists of five general contexts, so-called domains:

1. **Simple** – everything is ordered and stable. People simple follow the best practices described in the standard procedures.
2. **Complicated** – everything is ordered and quite stable as well. Experts have the freedom to use any of several applicable practices according to the considered scenarios and their personal experience.
3. **Complex** – situation is vague and unpredictable. Creative thinking and experiments are needed to find the appropriate solution.
4. **Chaotic** – unordered stressful conditions. Decision maker has to have enough power and intuition to reestablish control and get immediate positive effect.
5. **Disorder** – destructive state when not only the solution is unclear, but the situation itself as a whole is not understandable. Usually, Disorder can be resolved or at least explained on the higher level of view. Such a situation has to be guided into one of the other four domains as soon as possible. Disorder is not discussed in detail in this article.

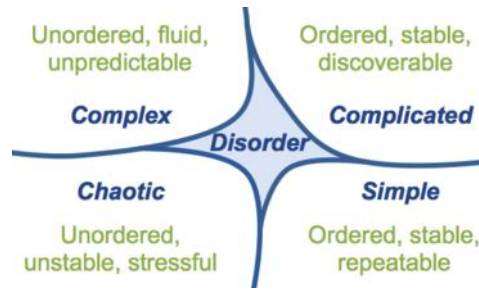


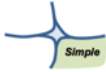
Figure 14: There are 5 Cynefin domains

Different people can interpret the same situation from the perspective of different domains. The stronger the importance of the issue, the more people seem to pull it towards the domain where they feel most empowered by their individual capabilities and perspectives. (Kurtz & Snowden, 2003)

	SIMPLE	COMPLICATED	COMPLEX	CHAOTIC
Situation	Ordered, stable, repeatable	Ordered, stable, discoverable	Unordered, fluid, unpredictable	Unordered, unstable, stressful
Main focus	Efficiency	Expertise	Creativeness	Rapid response
Practices	Apply best practices	Apply any of several applicable good practices	Discover novel practices	No common practice; experience and intuition are the key factors
Main approach	Standard operating procedures	Scenarios planning	Adaptive thinking	Crisis management
Principal actions	Ensure that proper processes are in place	Create panels of experts	Make experiments; simulate, create prototypes; use methods that can help to generate ideas	Take immediate control actions to reestablish order, look for what works instead of seeking the right answers
Optimal processing	Assess facts -> Categorize them -> Follow established procedures	Assess facts -> Analyze them -> Choose any of the possible solutions	Experiment -> Assess results -> Choose solution based on observations	Act on intuition -> Observe the immediate effects -> Decide to follow the direction if the action is effective, or rapidly respond with another option if it is not
Quality	Good results every time	High certainty in quality of results	Medium or low certainty in quality of results	Uncertainty in quality of results
Success	Easy replication of successful projects	Every successful project increases the probability of success in the next	Success in one project provides experience but does not increase the probability of success in the next	Success in the project does not depend on the success or failure in the others

Table 2: Cynefin domain describes the context of the situation and advises suitable strategies for dealing with it.

To clarify the idea of the Cynefin framework, let's consider the *example* of the particular problem resolution. Suppose that users complain of the application performance. In various contexts, the problem is resolved differently.



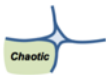
After analysis of symptoms, problem was classified as well-known. Administrator identified the procedure to follow. That is the **Simple** domain. Resolution is straightforward - according to user guide, specific configuration option has to be changed.



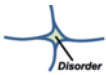
After analysis of log files, experts detected that the problem is in the storage subsystem. They developed a couple of alternative options with similar effects that could help. That is the **Complicated** domain. The resolution is to choose one of the possible equally good practices to optimize data volumes on the disk array and follow it.



Symptoms do not point to the source of the issue. Administrator has no idea what can resolve the problem. That is the **Complex** domain. Direction for resolution is to conduct some experiments and choose the solution based on observations. In effect, the administrator is using a trial-and-error method. First attempt is to add indexes into the database. Second is to optimize access to the data volume where DB tablespaces are located. At the same time, the administrator opens the service request for a vendor support.



The problem with performance appeared suddenly at the end of the quarter, a time when the application is needed for the success of the entire company. The problem must be resolved urgently, placing Administrators under pressure. They do not know the source of the problem and panic. Problem-solving becomes **Chaotic**. The resolution is to elect a leader from among the administrators and give that person required authorities. The first task is to move from Chaos to Complex domain. The leader should define the most probable causes of the problem and begin trial-and-error experiments as in the Complex domain. If probable causes are not obvious, the leader can start with “cheap” solutions; those can be quickly checked with minimal effort.



Senior administrator is on vacation. Junior specialist receives complaints about low application performance. Because of low-level of experience, he doesn't understand the problem. Under pressure from users, junior administrator is scared and totally paralyzed. That is typical **Disorder**. Problem cannot be resolved with the current level of expertise. The only way is to ask somebody experienced for help. The expert should analyze the situation and offer the solution in any other Cynefin domain according to the nature of the problem.

Every domain requires its optimal approach to diagramming. Consideration must be given to things such as characteristics as diagram's type, focus points to pay attention to, scale of changes needed to modify the existing templates, subject of these changes, how to encode information, required level of consistency, reusable blocks that can be applied to create new diagrams, structure flexibility, layout density, possibility to decorate, and allowed level of noise. The article explains all these characteristics in detail.

	SIMPLE	COMPLICATED	COMPLEX	CHAOTIC
Type	eXplanatory	mainly eXplanatory	mainly eXploratory	mainly eXploratory
Focus	on one specific task	on several specific tasks	on potential possibilities	on the situation as a whole with the most important details
Changes	no or minimal changes are allowed	some major changes are expected	changes are the main purpose	major changes are possible
Subjects to change	attributes only	entities, relations, and their attributes	entities, relations, and their attributes	entities, relations, and their attributes
Encoding	common well-known encoding only	usually common well-known encoding	novelty encoding is allowed	common and new encodings are allowed
Consistency	consistency is strictly required	consistency is strictly required	consistency is expected	inconsistency is allowed
Building blocks	Ready-to-use diagram	templates	visual patterns	everything that helps to present a situation holistically and quickly
Structure	Rigid, clear structure is required	Flexible, clear structure is highly recommended	Flexible, clear structure is recommended but not required	vague structure is allowed
Density	tight is recommended	not very tight is recommended	loose is recommended	insignificant
Decorations	decorations are allowed	minimal decorations are allowed	decorations are not recommended	insignificant
Noise	no noise	some noise is allowed	noise is allowed	noise is allowed

Table 3: Create diagrams considering the features specific to the situation's context.

Several *examples* are presented and described below to explain the features unique to every domain.

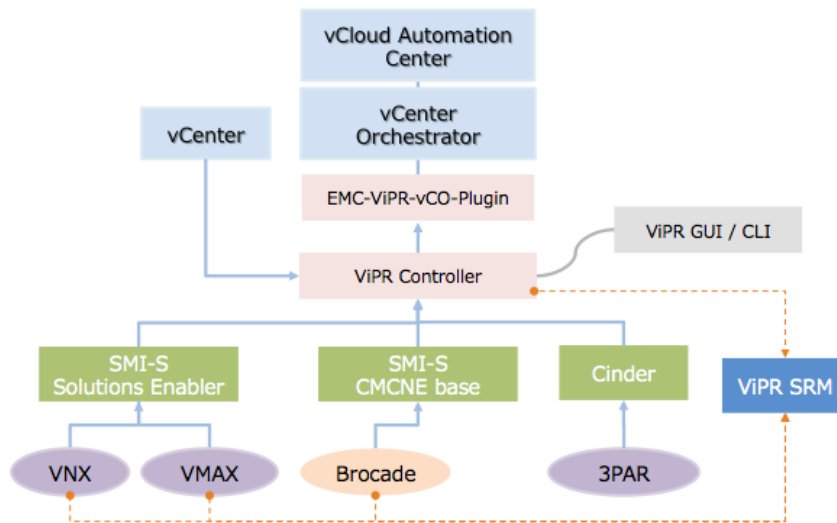


Figure 15: Diagram developed in the Simple domain; ViPR integration into vCAC / vCO, standard architecture diagram. Diagram can be used in different projects without any changes.

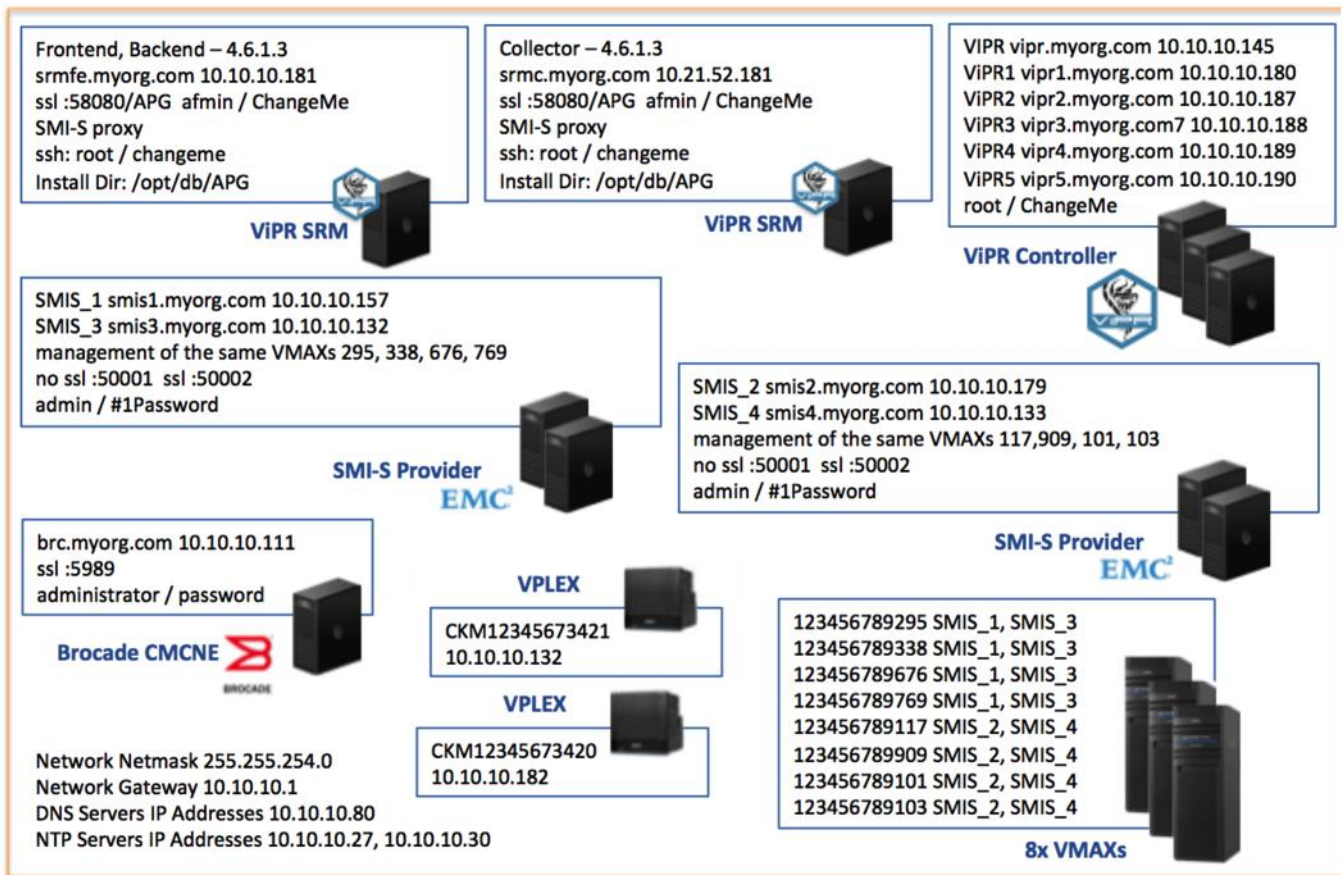


Figure 16: Diagram developed in the Simple domain; ViPR and ViPR SRM configuration parameters are presented. To use the diagram in the similar project, change IPs and credentials only.

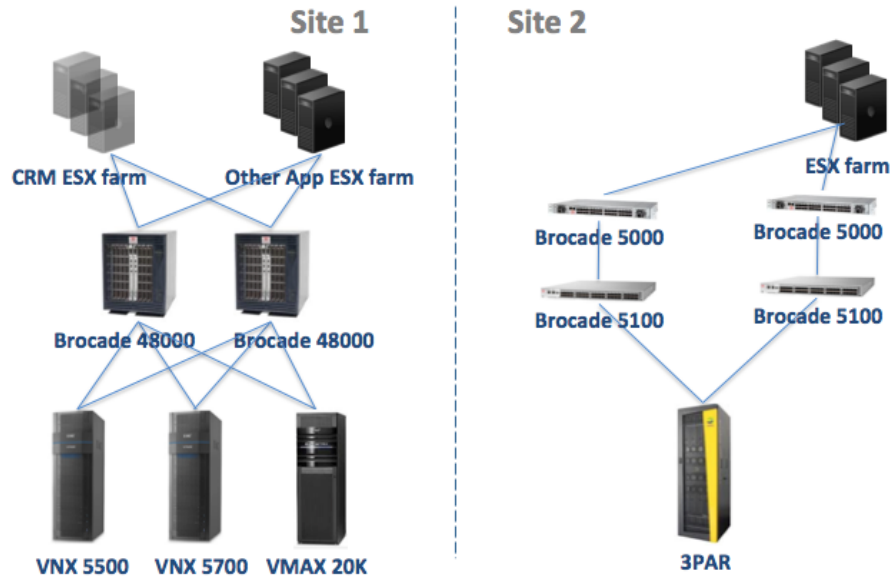


Figure 17: Diagram developed in the Complicated domain; Physical infrastructure is described. It was developed for a particular customer. In other projects, the standard entities, relations, and attributes are different, but the general approach remains the same.

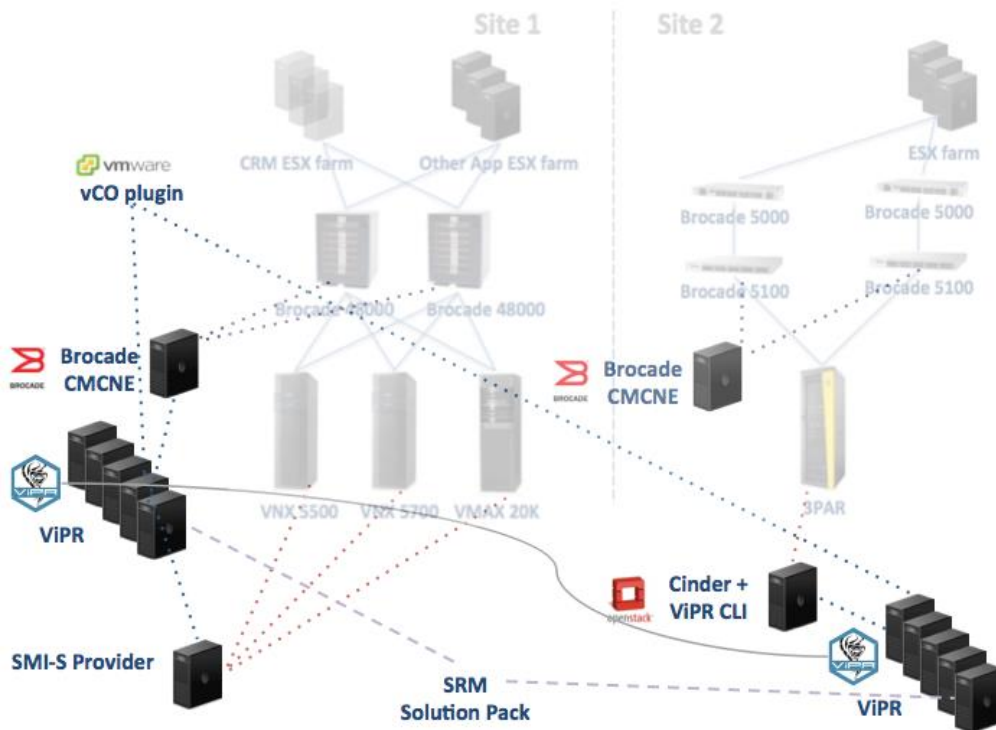


Figure 18: Diagram developed in the Complex domain; ViPR design prototype was quickly developed as a draft to discuss the pros and cons of the solution. Entities and relations are subject to change. The diagram is built on the basis of another one, previously created in the context of the Complicated domain.

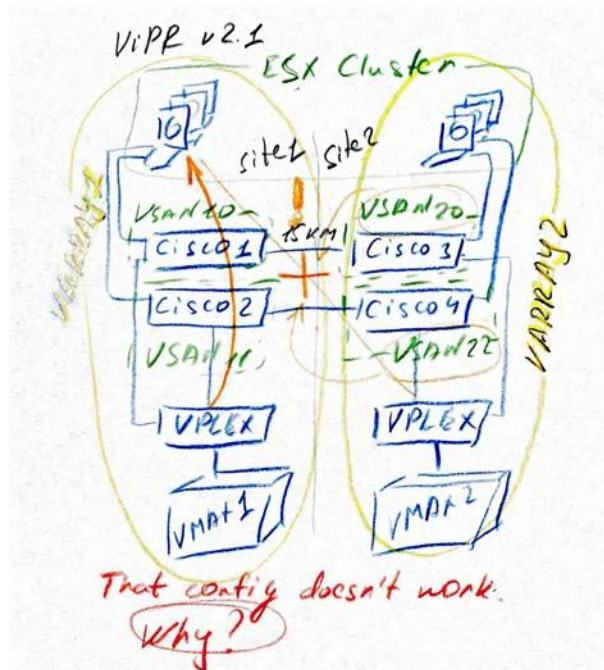


Figure 19 Diagram developed in the Chaotic domain; The main goal is configuration troubleshooting; ViPR can't provision VPLEX volumes to the stretched ESX Cluster. Diagram was created quick and dirty to reveal the issue with the VSAN isolation.

Boundaries between domains are not solid. We **can cross** them if needed. For instance, the diagram created in the context of Chaos can be the basis for prototyping in the Complex domain. Later, the Complex domain diagram can be transformed to a template and therefore move to the Complicated domain. The template can be adapted to the requirements of the particular infrastructure and became the ready-to-use diagram in the Simple domain.

Chaotic domain requires a holistic picture as soon as possible. The quickest way to create such a diagram is to take existing diagrams from other domains and combine them to present a wider picture. That is an example of **inter-domain transition**. Some other context-related movements are possible as well.

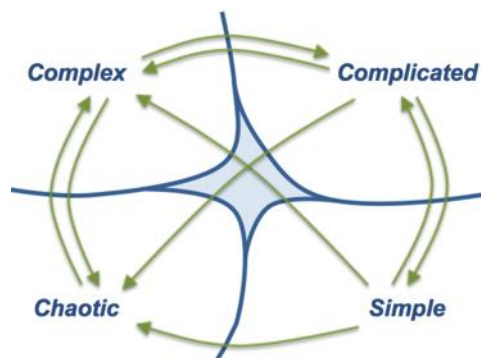


Figure 20: Diagrams can be transformed to satisfy the needs of other contexts.

Choose appropriate encodings

We can encode information by various types of visual properties or as it is called, **dimensions**. The most common dimensions are shapes and lines. In diagramming we can choose a size, orientation, position, line style and endings, color, brightness, contrast, icon, symbols, title, label, and many other dimensions. To decide which dimensions are applicable to satisfy the particular requirement, consider two key factors: attributes of appropriate type to be encoded and number of distinct values the reader needs to differentiate.

We have to use **types of attributes** to encode data. Some dimensions such as position, object length, or line weight are naturally **ordered** and can be used to encode ordinal data. Others, such as size and angle, are better suited to represent **quantitative** attributes. Shapes, colors, and connections are perfect to show **nominal** information. Of course, there are dimensions that fit several types of attributes. For *example*, text can be used to encode ordinal, nominal, and quantitative information.

The second factor to consider when choosing a visual dimension is the **number of distinct values** to differentiate. The reader has to be able to perceive and remember the information represented by the diagram. However, our brains are not equally good with different dimensions. We can effectively recognize only a few grades of brightness, but can easily differentiate a large number of shapes or line lengths and positions.

		Ordinal	Nominal	Quantitative	Values
23456	text	✓	✓	✓	infinite
○ ○ ○	position	✓	✓	✓	many
— — —	length	✓		✓	many
∠	angle	✓		✓	medium
○ ○ ○	size	✓		✓	many
■ ■ ■	brightness	✓			few
≡ ≡ ≡	line weight	✓			few
■ ■ ■	colour		✓		few
☺ ☹ ☹	icon		✓		many
≡ ≡	texture		✓		medium
□ △ ○	shape		✓		medium
□ □ □	enclosure		✓		many
□ □ □	connection		✓		many
— — —	line style		✓		few
— — —	line ending		✓		few

Figure 21: Attributes types and number of distinct values are the key factors to choose the right dimension for data encoding.

According to the research performed (Cleveland & McGill, 1984) there is a considerable difference in the accuracy of perception of different dimensions. The simplified and adapted for design diagramming version of **accuracy scale** is presented below.

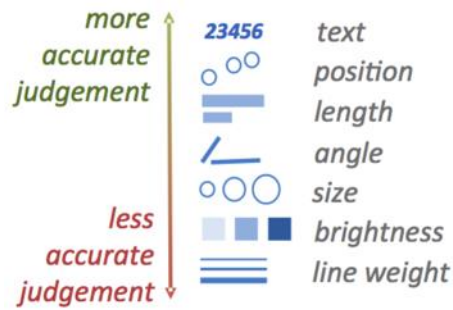


Figure 22: The more accurate judgement needed, the higher on the scale the dimension we should use.

In some diagrams, the goal is to present precise comparison of attribute values. In other cases, accuracy is not necessary because facilitating the perception of the larger pattern is all that is required. There are always many alternatives to choose the right dimension to represent your idea.

Explain encoding

Sometimes diagrams are created for internal design purposes and are never shown to anyone. However, in most cases we have to present the results of our work to the projects' stakeholders or share them with colleagues. Readers do not have the same mindset as yours, so they probably perceive the diagrams differently. To keep accuracy and efficiency of your diagrams appealing to a potential audience, **explicitly describe the encoding principles** we used. Titles, labels, and legends are the most common ways to define the meaning of the diagram and its elements.

When readers begin to use the diagram, they need to understand what to expect from it. The **title** of the diagram should point clearly to the subject being presented.



Figure 23: Every diagram should have a clear title; **Bad** example: “Sites location”. **Better**: “Distance between Site1 / Site2 and Site3 is about 240 miles”. **Best**: “Replication between Site1 / Site2 and Site3 is asynchronous due to the long distance.”

Use a **legend** to explain dimensions such as icons or colors. A good legend requires time to define and create, but that effort is rewarded by diagram clarity. If you consistently follow the established rules to encode information, the audience learns to read and understand your diagrams quickly. In a consistent set of diagrams, once you create a legend, it can be placed on the first diagram. If the legend is quite extensive, you can make a separate diagram just dedicated to the legend description.

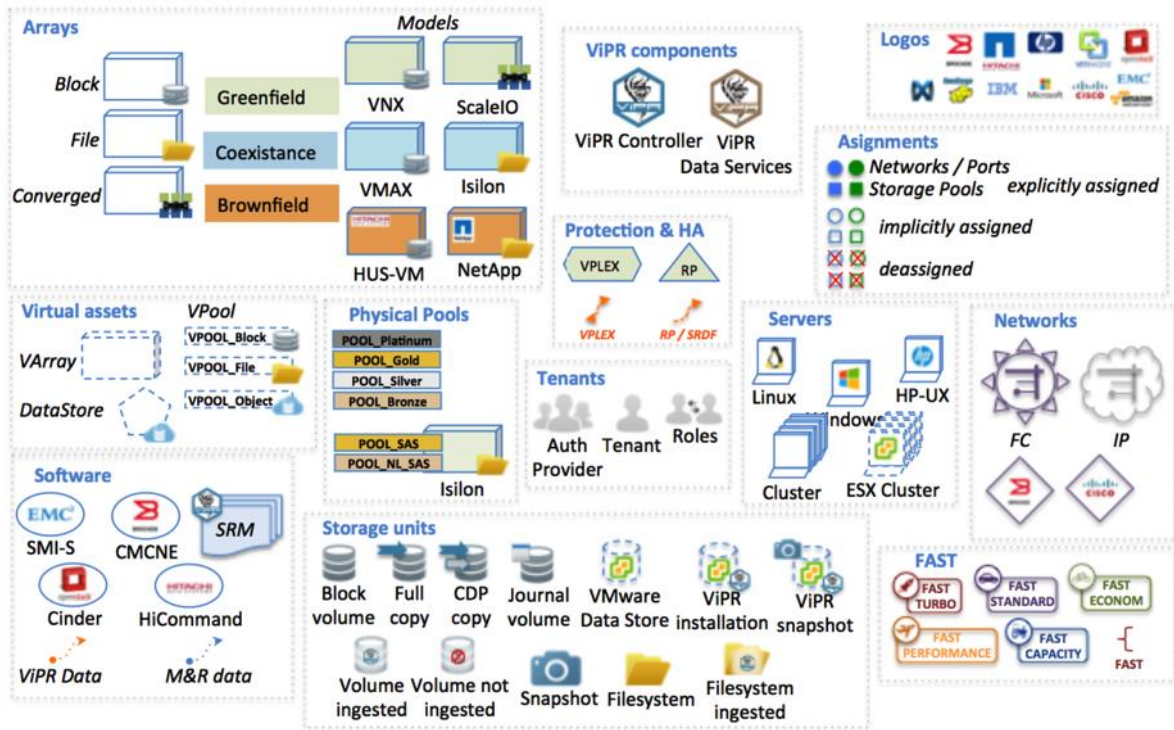


Figure 24: Legend used by the consistent set of design diagrams; ViPR design diagramming conventions

When you have fewer entities and relations or a less dense layout, you can use direct text **labels** to represent values and attributes of every item. Labels used for many densely located items can add visual noise. In such cases, designers should combine the general legend with labels for the most important entities and relations.

Be consistent in patterns

Upon discovering a visual image, the brain analyzes it in terms of primitive **shapes and colors**. Next, **unity contours and connections** are formed. As well, **distinct variations** are segmented. Finally, the mind attracts active **attention** to the significant things it found. That process is permanently running to react to similarities and dissimilarities in shapes, positions, rhythms, colors, and behavior. It can reveal **patterns** and **pattern-violations** among the hundreds of data values. That natural ability is the most important thing used in diagramming.

When readers notice the patterns, their brain assumes that they are intentional. The same goes for pattern-violations. Readers expect that violations are meaningful, whether it is planned or not. Therefore, we should always ensure that all similarities and dissimilarities support but not destroy the idea of our design.

To avoid implying unintentional meaning, designers have to follow the fundamental rule - **be consistent in methods to encode information**. That means that the same dimensions and types of attributes must be used to encode related information in individual diagrams as well as across a set of diagrams. Check and fix if your diagrams have unintentional patterns or violations in shapes, colors, line types, and other dimensions you use.

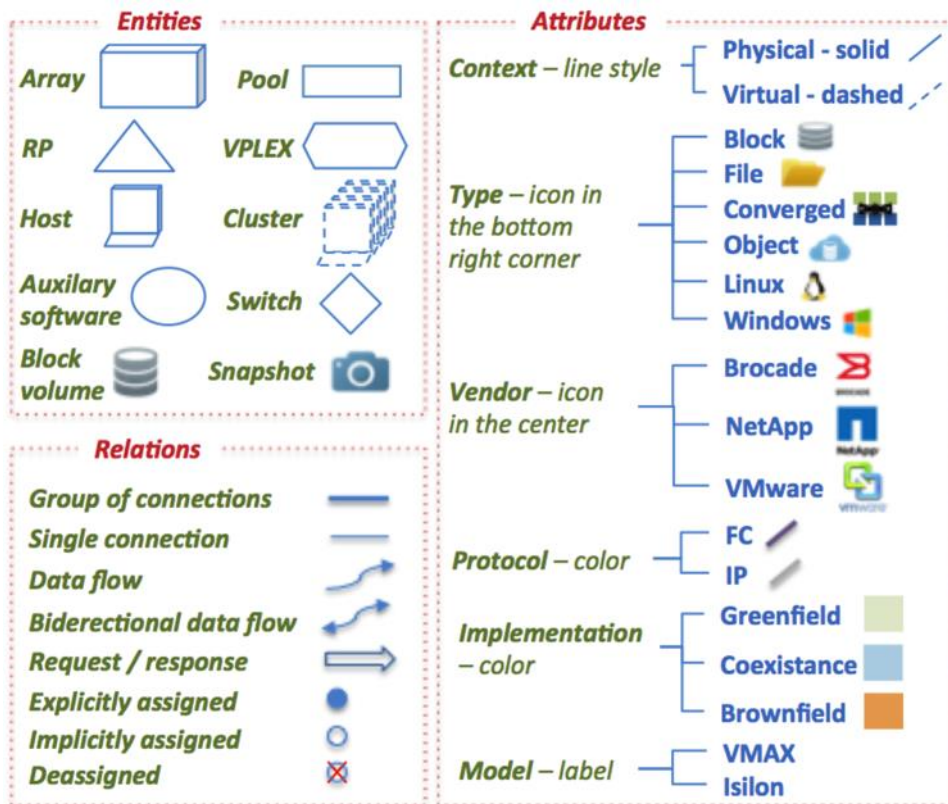


Figure 25: Rules defined for information encoding simplify the task of a consistent set of diagrams development; Encoding agreements used in ViPR design diagramming.

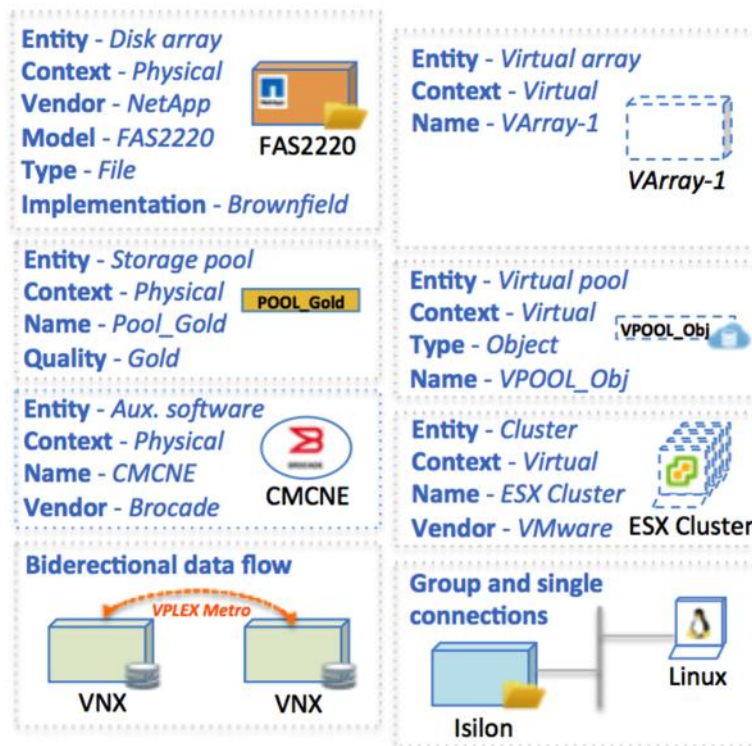






Figure 26: Entities, relations, and their attributes have to follow defined rules of encoding; Examples of objects used in a design of ViPR solutions.

Templates significantly help ensure diagram consistency in diagrams. Structure and pattern are already created. We just need to apply them to correspond to the particular requirements. A good template also decreases the risk of violating established patterns.

Pattern consistency is strictly required in the Simple  and Complicated  contexts. In the Complex  domain, consistency is recommended. In the Chaotic , some inconsistency is acceptable as the price of faster decision-making.

The following sections discuss the common methods of pattern creation via unity of different items and pattern violations using methods to distinct variety methods.

Use unity methods

According to **Gestalt** psychology, human perception tends to organize information into groups or unified wholes, rather than parts. There are five general methods used in diagramming to present several elements as a unity:

- **Proximity**

The distance between entities affects how we perceive and interpret it. We see entities that are close together as one unit and, vice versa, those are distant from each other as separate and unrelated.

- **Similarity**

We tend to see elements similar in size, color, shape, orientation, brightness, and some other dimensions as one group.

- **Good Form**

The principle of good form is the tendency to group together forms of analogous shape, size, color, etc. That principle covers likeness of visual dimensions in a wider perceptual perspective than Similarity. Therefore, it is more subjective and difficult to notice.

- **Enclosure**

Parts enclosed by a boundary are perceived as a group. That unity method works even if the boundary is incomplete or partially hidden by other objects.

- **Connection**

Entities are perceived as a group when they are connected by a line or common edge. That method is extremely powerful. It can override other grouping methods, such as Proximity and Similarity.

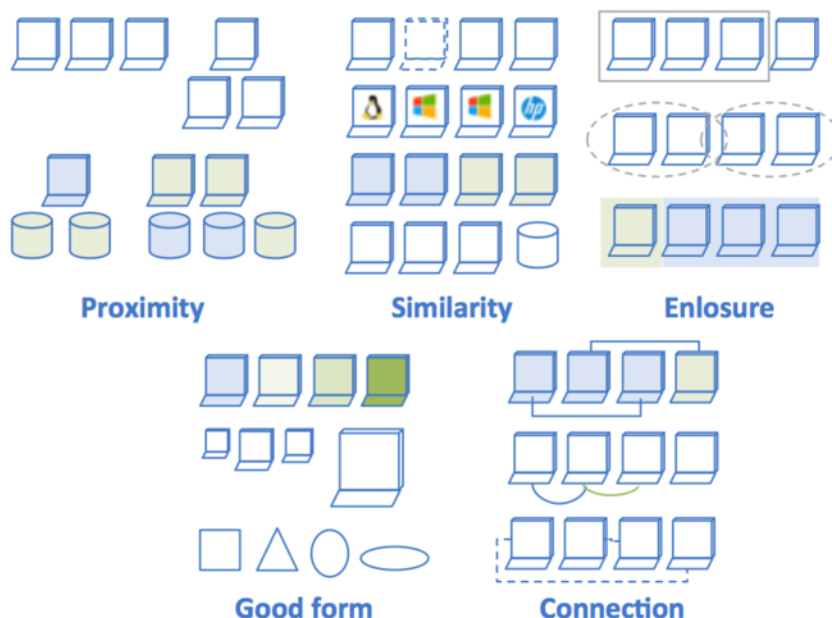


Figure 27: There are five general methods of grouping.

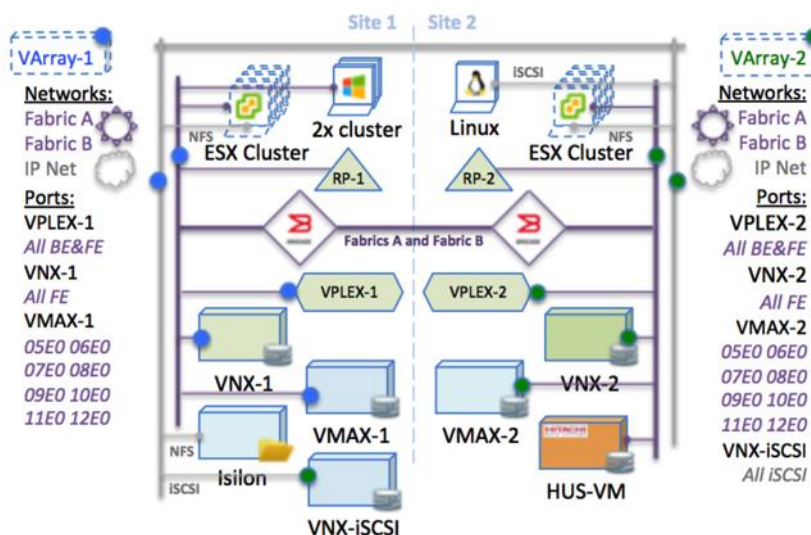


Figure 28: Similarity grouping; Networks and ports marked by blue circles are members of the VArray-1, green associates corresponding networks and ports with the VArray-2.

Use variety methods

To more easily find the particular entity or relation easier, make it distinct from the background and other elements. The most popular methods are based on these dimensions: **Colors**, **Texture**, **Contrast**, **Brightness**, **Orientation**, **Position**, **Enclosure**, **Size**, **Shape**, **Text**, **Icons**, **Pointers**, and **Symbols**.

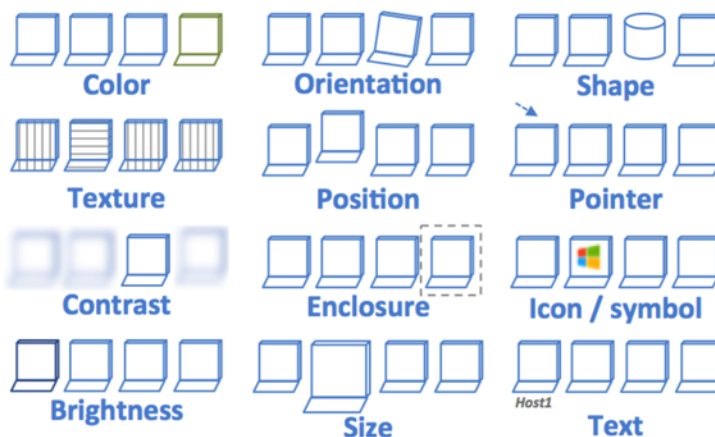


Figure 29: There are twelve common methods to make the element distinct.

Variation methods are not equally strong. In general, the most powerful effects are based on **color**, **orientation**, **size**, and **contrast**. For *example*, a bigger difference in color attracts more attention than does brightness. Use stronger variety methods before weak ones.

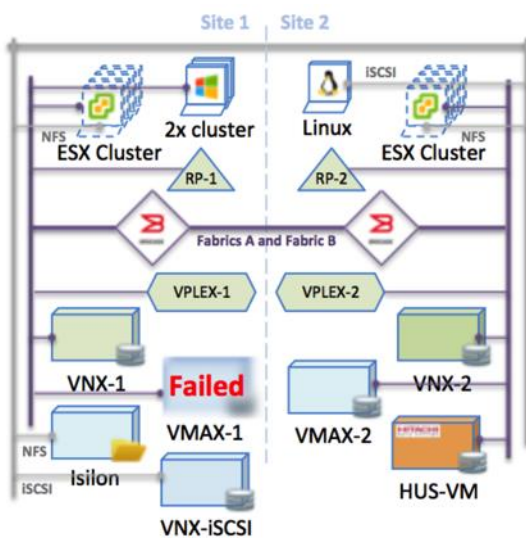


Figure 30: Redundant encoding makes elements more distinctive; To pay attention to the VMAX-1 fail we use contrast and colored text.

For maximum effect, the entity should be the **only** one in the diagram with the particular visual characteristic, i.e. it might be the only item that is colored red; everything else is black and white. To highlight the element on the visually complex diagram, apply a dimension that is used **least**. For *example*, if a text has the same font size, increase the size to attract attention. To maximize element distinction, encode it by **several** visual dimensions (such as size, orientation, and color). Such **redundant** encoding is discussed in detail later.

Make progress iteratively

Design process is **nonlinear**. At some steps, it might be necessary to return a little bit back or even start again from the beginning. As long as you analyze the situation and learn from the mistakes of the previous iterations, the overall progress moves you toward the final goal. The approach described below is not a step-by-step process, just a navigational tool.

There were three general phases in the design process:

- **Collect**
- **Ideate**
- **Reflect**

Down-to-up **CIR** stream is sequential. Reverse jumps can be from any to any phases.

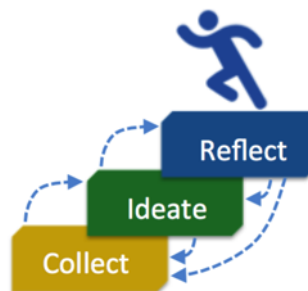



Figure 31: Three phases of diagramming are iterative.

Collection is a period to determine expected outcome, define goal statement, and collect / clarify information. There are plenty of methods to collect data, from face-to-face interviews to specialized tools. When information is needed but unavailable for any reason, realistic assumptions should be declared and confirmed by the stakeholders.

Ideation phase is intended to gain insights and make concepts. Big and small Ideas are generated and polished here.

This stage is not about avoiding mistakes; rather, it's about **exploring as many mistakes as possible**. (Stickdorn & Schneider, 2012) The relative cost of an additional iteration during the Ideation stage is quite low compared to the price of failure after the launch of the concept. Therefore, it is better to make mistakes as early as possible.

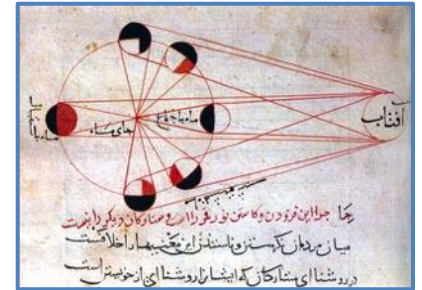
This phase should be creative in the right meaning of this word. According to (Higgins, 1994) "Creativity is the process of generating something new that has **value**. There are many original ideas and concepts, but some may not have value and hence may not be considered creative". Remain pragmatic and focused on the goal and expected value of your diagrams.

 The purpose of the **Reflection** phase is to make intangible ideas evident. We create conceptual mockups of diagrams and their detailed prototypes as well as present the final result. Elaborate designs usually require more iterations between Reflection and Ideation phases.

A common mistake in design is trying to create the final version of the diagram in one attempt. Trial-and-error is the right approach for diagramming. Thus, **prototypes** are an essential part of the process. They help to exclude bad concepts and focus on those most promising. Even if you spent significant time to create a diagram but found that it still has conceptual weakness, do not regret the effort you spent. It is better to consider the unsuccessful version as a prototype and not keep it but just make the next steps for improvements. From a strategic perspective, such a decision is much “cheaper” than using improper an eXploratory or eXplanatory tool.

Section summary

- Main principles of efficient diagramming are:
 - Define the goal
 - Define the context
 - Choose appropriate encodings
 - Explain encoding
 - Be consistent in patterns
 - Use unity methods
 - Use variety methods
 - Make progress iteratively
- A goal statement includes an action verb, object of action, and contextual clarifier.
- Cynefin framework contains the Simple, Complicated, Complex, and Chaotic domains. It helps to identify the context and advise a suitable strategy to create the diagram.
- Critical factors for encoding are type of attributes and number of distinct values to differentiate.
- Titles, labels, and legends explain the meaning of the diagram.
- Be consistent in methods to encode information.
- Methods to group elements are Proximity, Similarity, Enclosure, Good Form, and Connection.
- Methods to distinct elements are Colors, Orientation, Position, Enclosure, Size, Shape, Text, Icons, and Pointers.
- Design process is iterative. Three phases are Collect, Ideate, and Reflect (CIR).



Lunar Eclipse diagram; Abu Rayhan al-Biruni, 1019

4 How to make diagrams better?

Use different visual channels

Our brain processes colors, brightness, shapes, line style, position, attached icons, and text separately. Thus, these dimensions can be considered as different visual channels. Information presented in one channel (for example, color of the entity) does not interfere with data expressed in another (for example, size of the entity). We can use separate channels to present different attributes of items so that they are visually distinct.

Encoding is called **redundant** when different visual channels are used to represent the same information. Redundant encoding is an efficient trick that helps to understand information from diagrams faster, easier, and more accurately. For *example*, relations represented by lines can be encoded by different endings (dots, arrows), line style (solid, dashed), weight, or color. To make redundant encoding of an entity's shape, consider its position, icon, or text label.

The less common the visual form, the more redundancy we should include explaining novelty we use.



Figure 32: Redundant encoding; 1- shape is redundant to the text label. 2 - icon to label.

3 - icon, text, and color. 4 – shape that according to the convention means Storage Provider combined with Hitachi logo are redundant to the text label. 5 – one- and two-ended arrows are redundant to their colors.

Make diagrams compatible with the reality

To decode information easier, align it with the reality in perspective of both the physical world and cultural conventions. Some things have particular colors, are larger or heavier than other, or are associated with the specific place. If your encoding is not compatible with these properties, readers may wonder why things do not look like they are expected to. Consequently, their auditory is forced to spend extra efforts decoding.

Consider a couple of *examples*. If one site is geographically located in the west and another in the east, you should place the first site on the left part of the diagram and another on its right. As well, since

people consider software as lighter than hardware, software components of the solution should be placed higher than hardware.

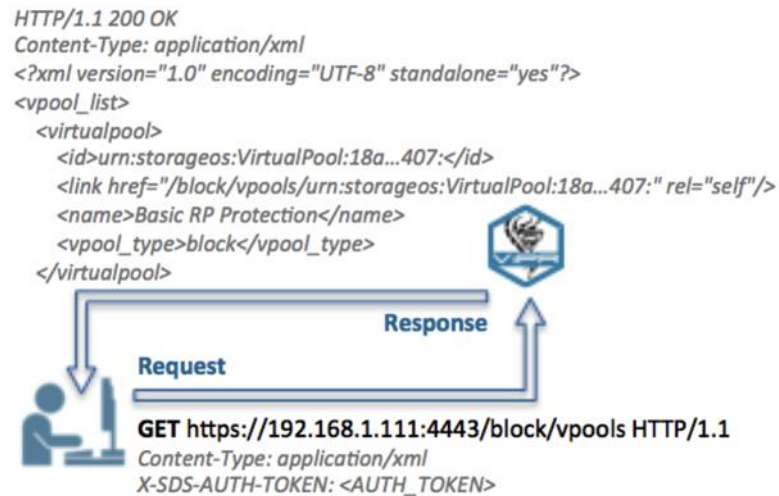


Figure 33: Bad example of compatibility; Request and following Response contradict the natural way to read data from top to bottom.

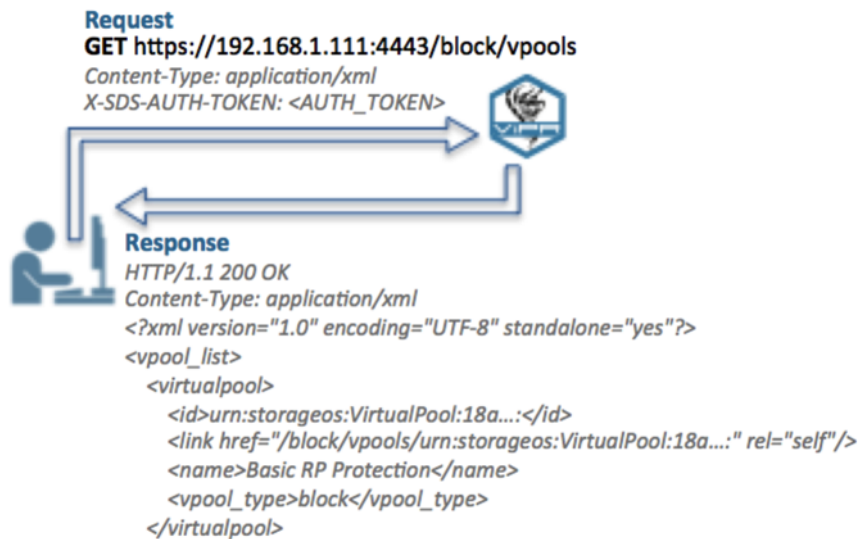


Figure 34: Good example of compatibility; Response naturally follows the Request.

As we discussed, colors are not naturally ordered. However, there are **social conventions** about meaning of colors. For instance, green and red have strong meanings for bad and good, or stop and go. There are many other cultural conventions about things such as top versus bottom, bright versus dark, square versus round, and big versus small. Avoid conflicts of such conventions with expected interpretation of the underlying idea. Moreover, use them when possible to reinforce an expression.

Compatibility with the reality should be used considering other factors. For *example*, to encode products of different vendors, their branding colors can be used. Blue can be naturally associated with EMC, Red with HDS, green with Violin Memory.

However, if we represent products of EMC, HP, Dell, NetApp, and IBM on the same diagram, the idea of color compatibility would be wrong. Five hues of blue are the worst candidates for encoding of nominal data.



Figure 35: Good and bad examples of storage vendors brand colors' compatibility; Small number of different colors much easier to differentiate than many hues of blue.

Define structure

In theory, we always have to begin diagram creation with a development of the structure. Ideally, that is the perfect approach but it rarely work on real projects. You can begin your design with either structure or collected data and change both during the iterative process. However, defining the structure in the early stages of creation helps keep the diagram organized and in perspective to the particular design goal. Structured pattern is easier to find and recognize.

Consider several factors of diagram structure:

- Diagram can be structured according to one or several **characteristics**. Examples are *Locations* (sites, logical areas), *Types* of components (hosts, storage), and *Hierarchy* or *Order* of parameters (subordinated roles, storage capacity).
- Characteristics can be **independent** or **nested**. For example, hardware components of the solution are located in different places. In that case, every *Site* can be considered as the primary characteristic and *Type* of equipment as the secondary characteristic nested into the *Site*. Many levels of nesting are possible.
- Every characteristic can have one or several **regions**. For example, if a considered characteristic is *Location* then every particular site is a separate region.
- Regions can be **nominal** or **ordinal**. Nominal regions should be placed considering a compatibility with the reality. Position ordinal regions in the expected order.
- Regions usually placed in **vertical** or **horizontal** fashion. That is not strictly required but easier to design and perceive.
- Regions can be **functionally symmetrical** or not. For instance, if two sites are equal in perspective of their roles, they are functionally symmetrical. When Site1 is production and Site2 is devoted for disaster recovery, they are functionally non-symmetrical.
- Structures can be **visually symmetrical** and non-symmetrical (discussed in the next section).

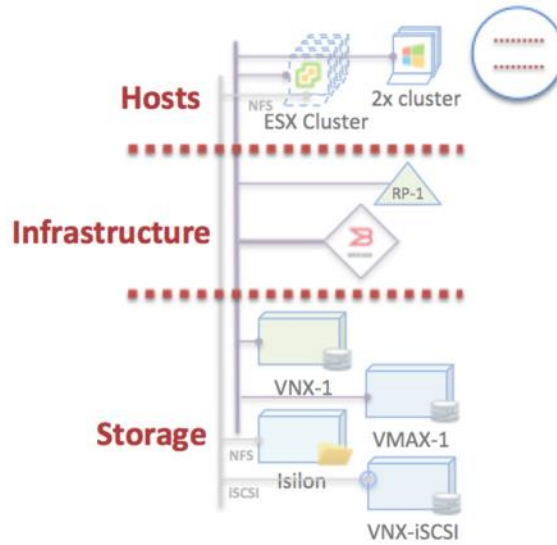


Figure 36: Vertical 3-regions structures; Physical components are layered as Hosts, Infrastructure, and Storage.

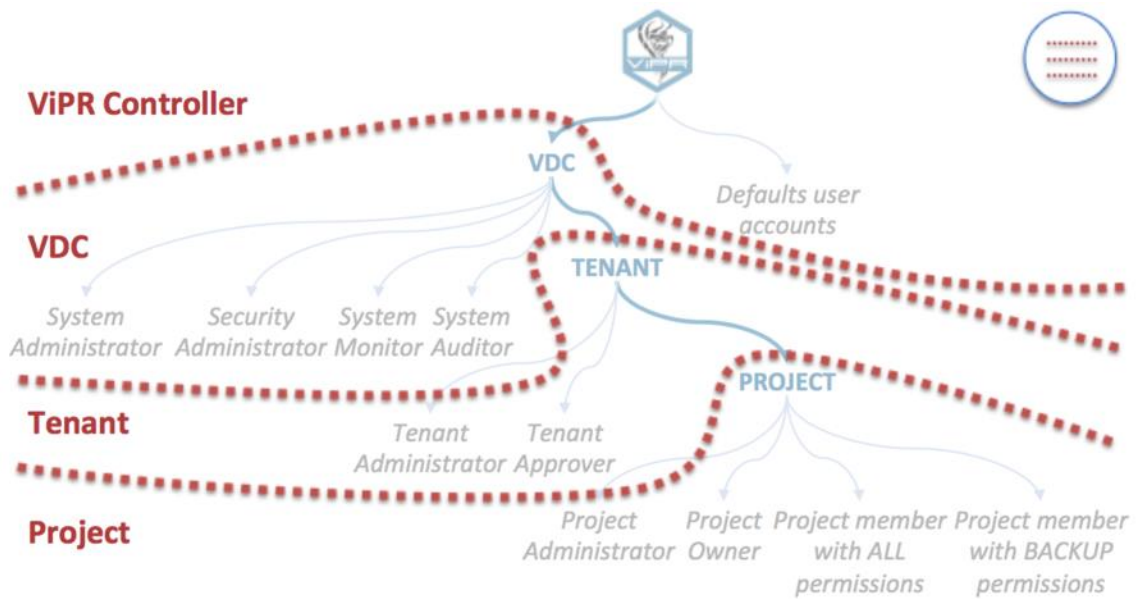


Figure 37: Vertical 4-regions structure; Roles in ViPR are divided on ViPR Controller, VDC, Tenant, and Project.

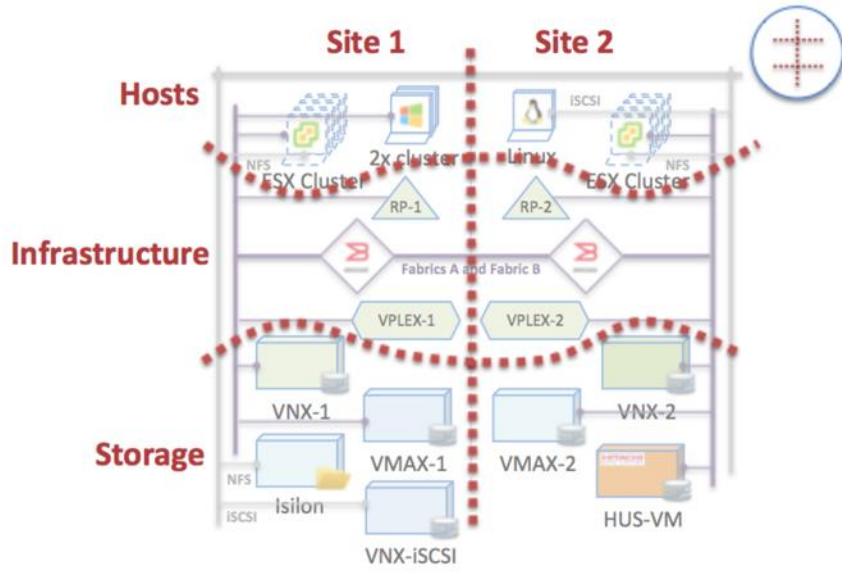


Figure 38: Horizontal 2-regions functionally symmetrical and vertical 3-regions non-symmetrical structures; Both sites are in production. Physical components of the solution are divided into groups of Hosts, Infrastructure, and Storage.

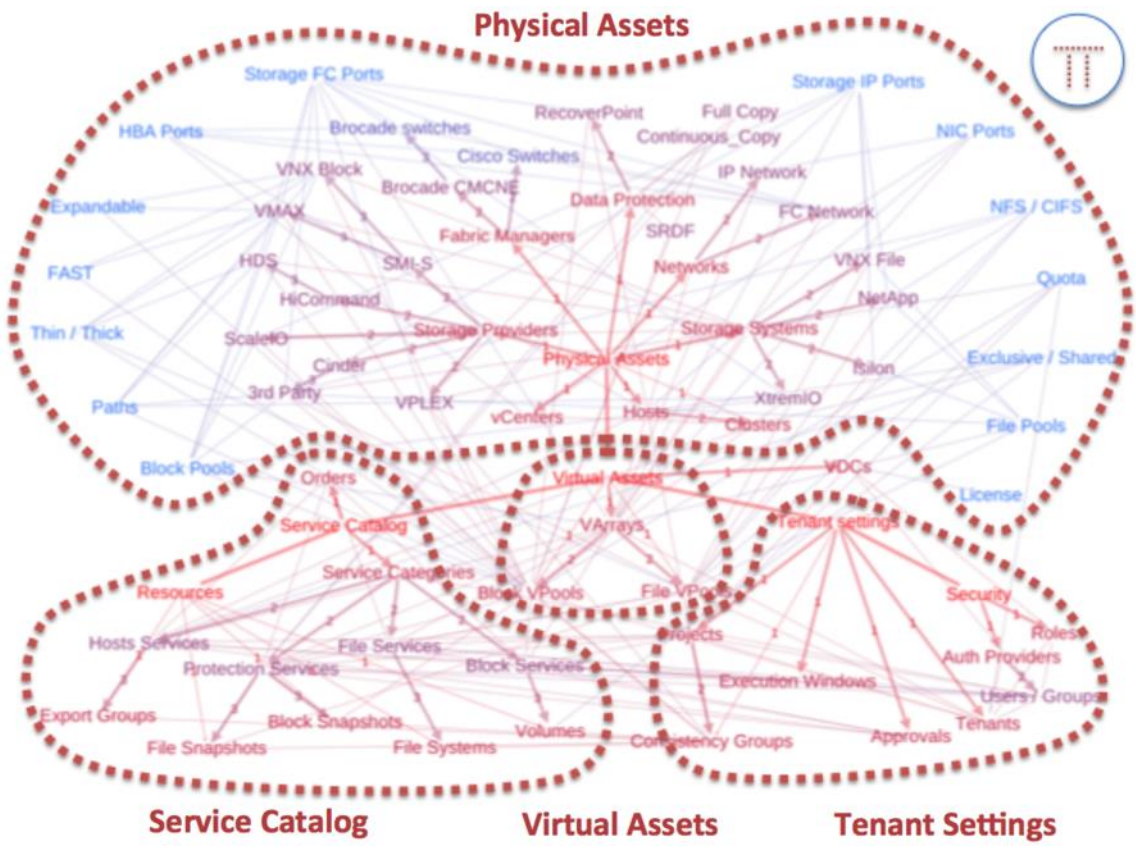


Figure 39: Structure contains 4 regions; Main VIPR configuration blocks are Physical Assets, Virtual Assets, Tenant Settings, and Service Catalog.

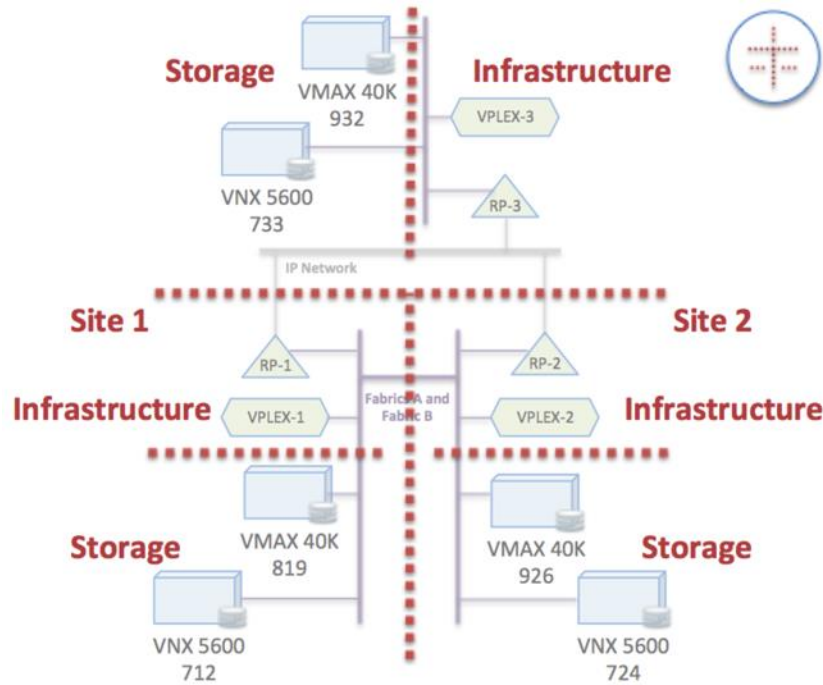






Figure 40: Two nested structures; Equipment is located in three sites. Infrastructure and Storage components are configured on each site.

Simple  Cynefin context requires clear and rigid structure. Diagrams created in the Complicated  domain should have a flexible structure. Recommendations to be structured in the Complex  domain are even less strict. However structure is still helpful there. In Chaotic  situations, structure is not a high priority recommendation and can thus be quite vague.

Create symmetrical visual patterns and structures

Human brains better recognize symmetrical visual patterns. Hence, designers try to follow the reflectional, scale, or translational symmetry when possible.

- **Reflectional** symmetry represents situations when elements of the diagram are mirror images. In diagramming, this type of symmetry should be considered wider than just strict mathematical definition. Brains perceive elements as symmetrical even if some of their parts (like icons or text labels) are not the same.
- An element has **scale** symmetry if it does not change shape when it is proportionally increased or decreased in its size. Scale symmetry is typically used to compare ordinal or quantitative data.
- **Translational** symmetry describes the diagram element of the group repeated without changing as shifted in the specified direction by a constant distance. That type of symmetry usually is the result of the copy-paste replication of diagram elements.

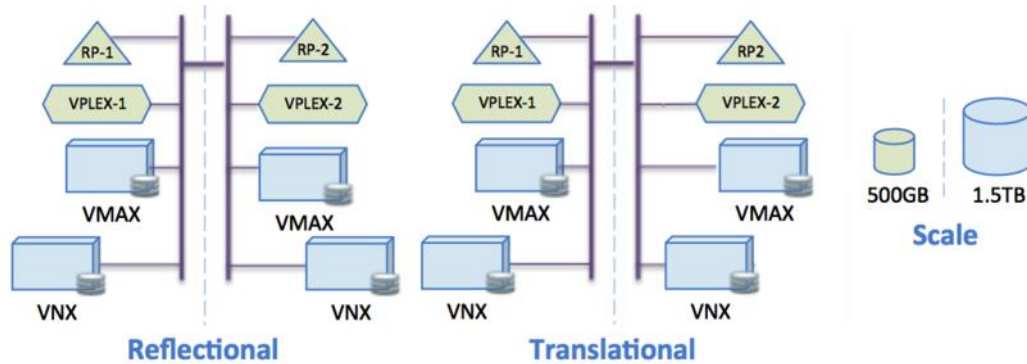


Figure 41: Reflectional and translational symmetries usually applied to groups of elements. Scale symmetry is better applicable to individual elements.

Patterns can be perceived as symmetrical not only considering shape of elements but their colors, line style, weight, and other visual dimensions.

Use whitespace

Whitespace – sometimes called **negative space** – is the part of the diagram left unused. Balance between positive and negative spaces is essential to create a functional and esthetic diagram. There are two types of white space macro and micro.

- **Micro** whitespace refers to small pieces of space between shapes, lines, titles, icons, and other elements next to one another. Individual elements in a diagram with more micro whitespace are more noticeable and easy to perceive, make a diagram less visually cluttered and, therefore, more attractive.
- **Macro** whitespace defines large volumes of space between elements. This type of negative space is the instrument to control proximity and remoteness between elements and form their unity or variety. Use it to make a group of objects or, vice versa, distinguish one entity from another. Macro whitespace helps to form the overall structure of diagrams.

Both micro and macro whitespaces influence overall **density** of a diagram. Micro space allows aligning and moving elements in some limits. Loose micro whitespace even allows adding numerous new entities. Loose macro whitespace provides room to add even more items and move them in wider limits if needed. Tight diagrams are easier to place in limited page or screen space and print.

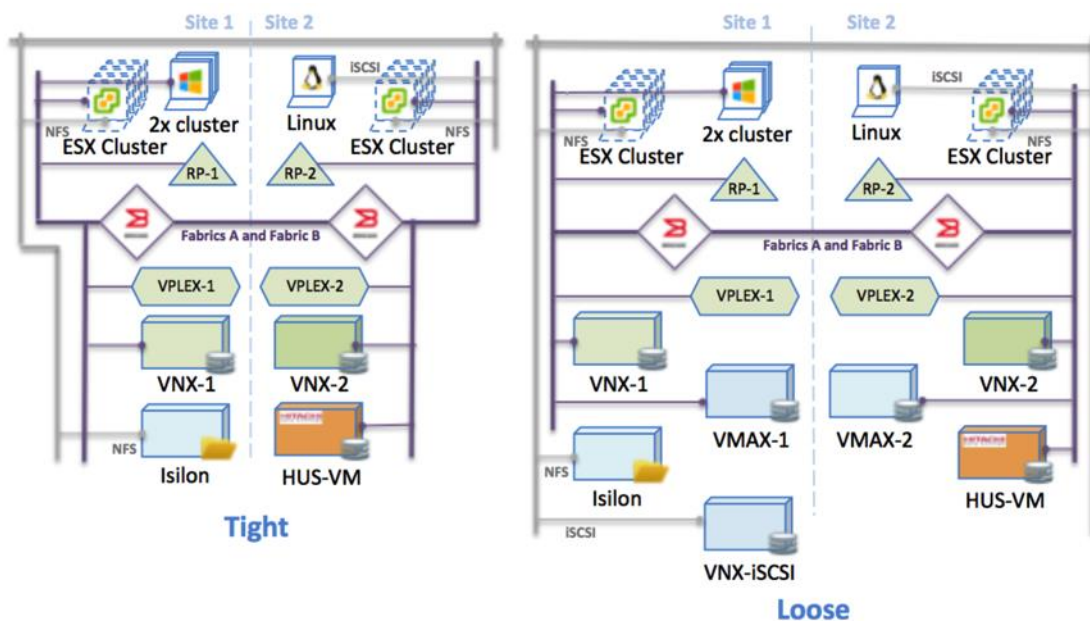






Figure 42: Tight diagrams are compact. Loose diagrams allow arranging elements and adding new entities; Two VMAXs and VNX-iSCSI were easily added to the loose diagram. The tight one does not have space for that.

Diagrams In the Simple  domain are usually quite tight. They are very eXplanatory and less density does not add more clarity. On the other hand, the overall space these diagrams occupy is important.

Complicated  and Complex  domains diagrams require more space to play with placements and alignments of entities and relations during the Reflection phase of design. Chaotic  diagram can be combined with parts of different densities.

Create complex based on simple

As we discussed, in real life the process of design is very iterative. We cannot say that eXploration or eXplanation should always progress from simple to complex. However, it is much easier to create a diagram with fewer entities, relations, and attributes first and increase its complexity afterwards. Thus, at the beginning of the diagramming, decide if there is a possibility to create the diagram that could be the **minimalistic basis** for the creation of more sophisticated diagrams.

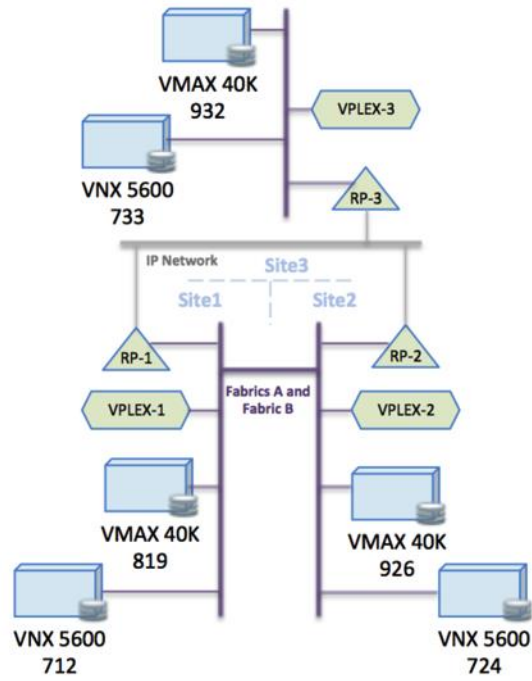


Figure 43: Diagram that is used as the basis for the coming design; Solution covers 3 sites. It is based on VNXs, VMAXs, VPLEXs, and RecoverPoint appliances.

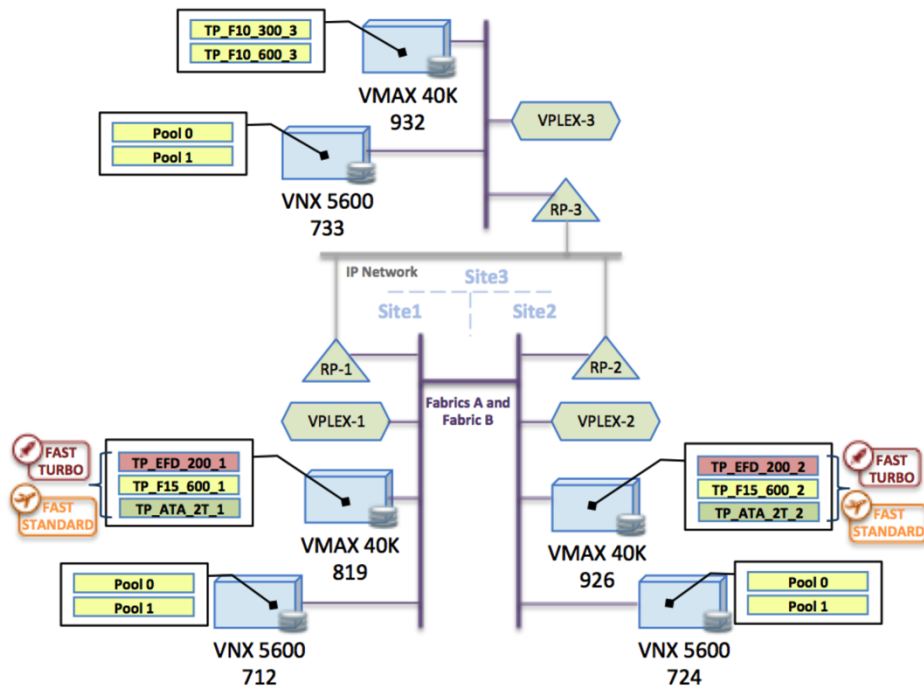


Figure 44: Configuration of the components is added to the basic diagrams; VNX / VMAX Storage Pools and FAST policies are configured.

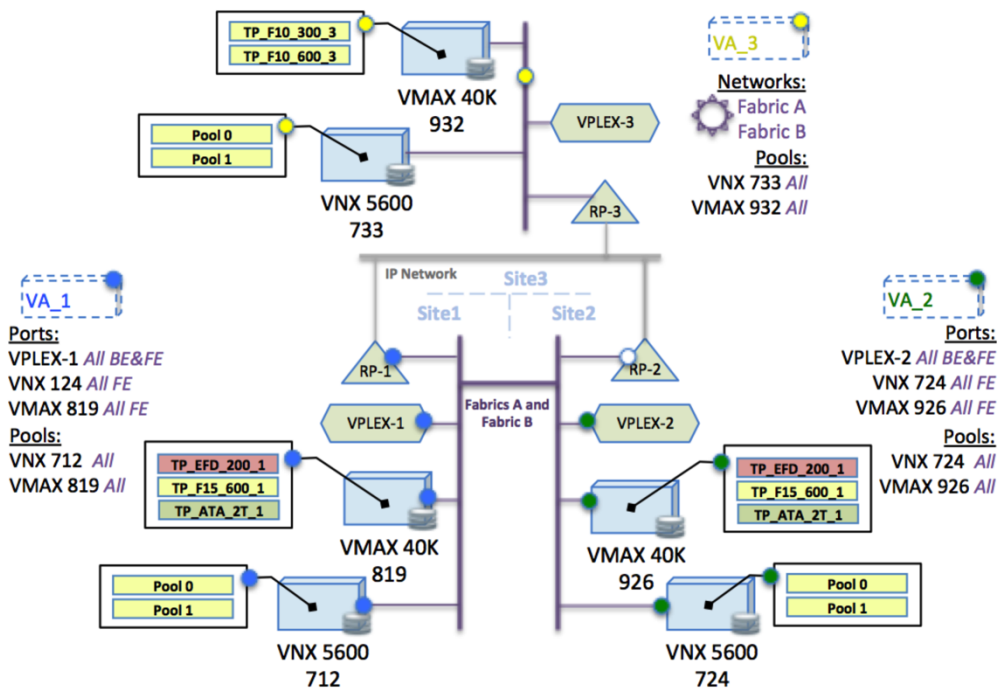


Figure 45: The next level of complexity is presented; Separate Virtual Arrays are configured on each site.

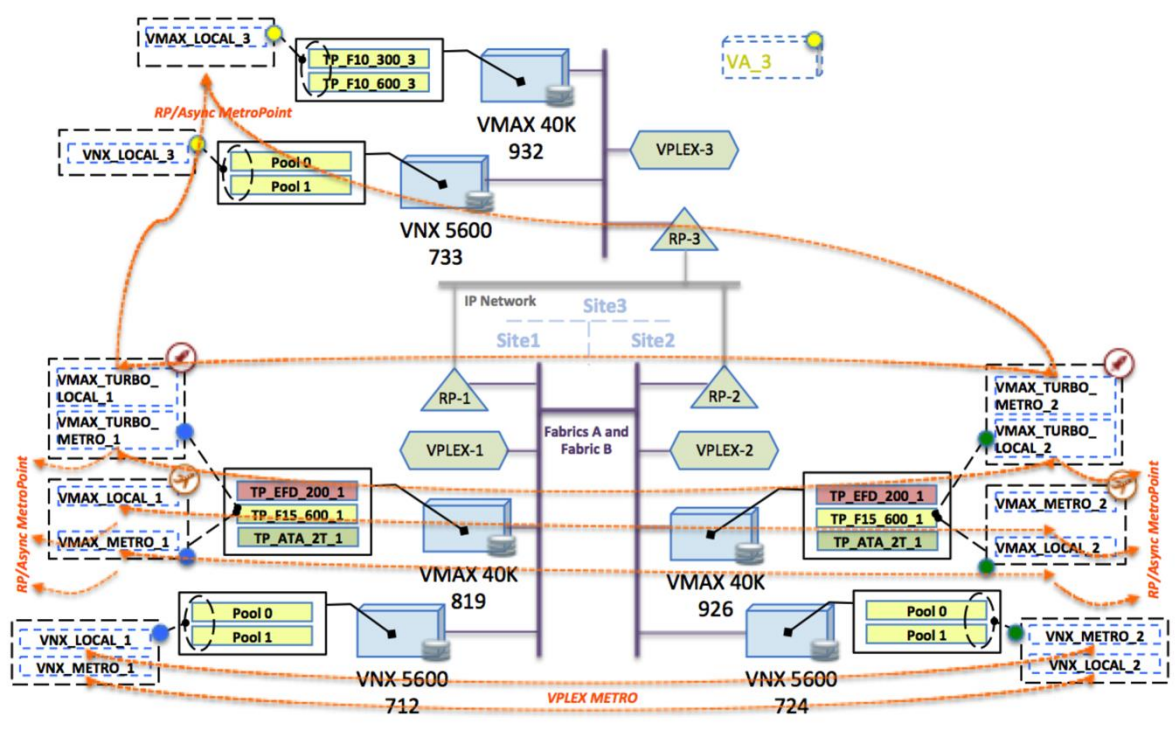







Figure 46: Some elements were added and deleted from the previous diagram to proceed with the design; Virtual Pools follows the FAST, VPLEX Metro, and RecoverPoint MetroPoint requirements

When creating the basic diagram, please consider consistency, density, and structure required by consecutive diagrams with more elements. If later diagrams reveal that you made consistency mistakes in the Simple  and Complicated  contexts, you have to fix the inconsistency in the primary and all following diagrams. In the Complex  and Chaotic  domains, some inconsistency is allowed so do not spend your time on inconsistency remedy there.

If you went wrong with the structure or density of the basic diagram, a fix for the whole set is required only in the Simple  context. In other domains, just adjust the structure and density according to your particular diagram needs.

It is important to understand that the number of elements on the diagram does not necessarily represent **complexity** of the of eXploration or eXplanatory idea. Therefore, the described approach just simplifies the process of diagram creation, not thinking progress.

Decrease noise

Usually, diagrams contain some noise – information unrelated to the diagram’s primary goal. Noise is decorations, redundant, and irrelevant data, unnecessarily emphasized and ambiguous icons, symbols, lines, grids, or labels. Every unnecessary element draws attention away from the central idea that the designer is trying to share. Noise reduces clarity by hiding useful information in a fog of useless data. You may quickly identify noise elements if you can **remove** them from the diagram or make them less intense and attractive without compromising the function.

The clarity of information depends on the ratio of relevant to irrelevant information in the diagram, a characteristic called **Signal to Noise Ratio (SIR)**. Good diagramming maximizes useful information and minimize noise, thereby producing the highest possible SIR.

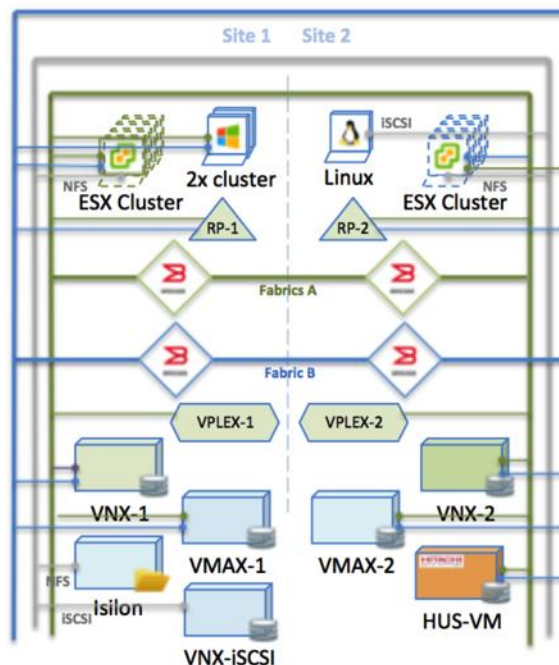


Figure 47: **Noise** requires more effort to analyze diagram; Redundant FC fabrics (A and B) do not offer any advantages for eXploration of the physical environment but draw attention from the elements of design.

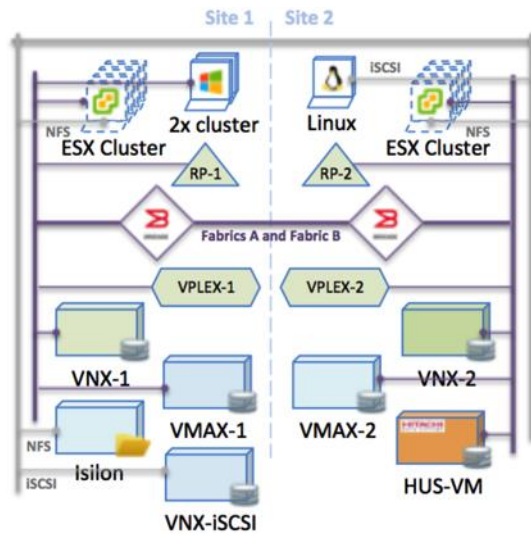






Figure 48: Clean diagram is easier to eXplore; Single SAN represents both fabrics.

eXploration is best done at a very **detailed level**. There may be much noise data, but you should get rid of it carefully so as not to miss something important. By contrast, eXplanation requires **cleaning up** all distracting or irrelevant information. You have to focus the reader's attention on the data that supports the idea you are trying to highlight.

Diagrams in the Simple  context must be as clean as possible. Diagrams in the Complicated  domain can contain some noise. Prototyping in the Complex  and quick & dirty approach used in the Chaotic  allow noise in these domains.

Make diagrams stylish and esthetically attractive

Does the minimum noise concept mean that we always must remove everything that is not absolutely critical to the design? Well, yes and no. Some decorations, redundancy, and unnecessary data also play their role. Stakeholders are people with their interests, subjective preferences, and emotional experience. So designers must reflect not only functional elements of their solution but emotional aspects as well. **Functional** aspects focus us on the goal to achieve, whereas **emotional** aspects are related to feelings and perception. They are critical for eXplanatory diagrams.

In diagramming, function has to be first. Facts and logical arguments are essential to explain the idea. However, stylish and esthetically attractive diagrams do that job even better. An additional emotional channel of information perception reinforces the total effect on sharing the designer's personal experience, enthusiasm, and solution elegance. Of course, functions and emotions must be **balanced**. Too much decoration makes diagrams excessively noisy. When we make cold minimalistic diagrams, we decline the extra possibility to utilize redundant explanatory channels.

Balanced use depends on the particular circumstance, audience, and diagramming objectives. High SNR is good as the general attitude, but not the rule to be strictly and mindlessly followed. It is just one approach among many to consider when creating diagrams.

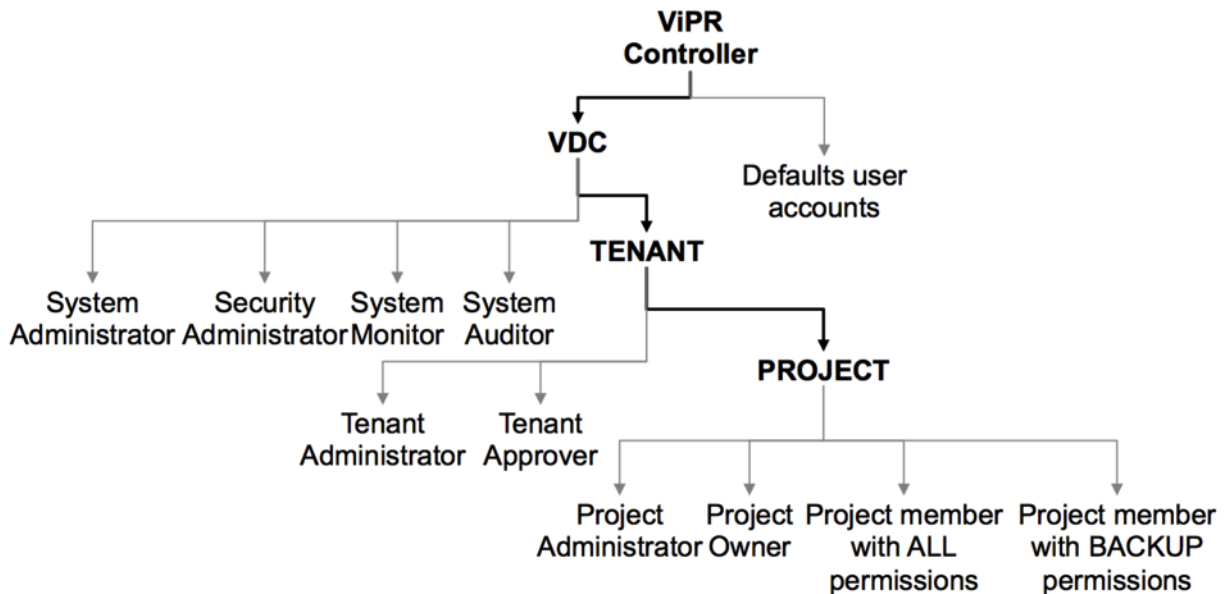


Figure 49: **Bad** example of a diagram esthetic; Diagram is boring and difficult to perceive.

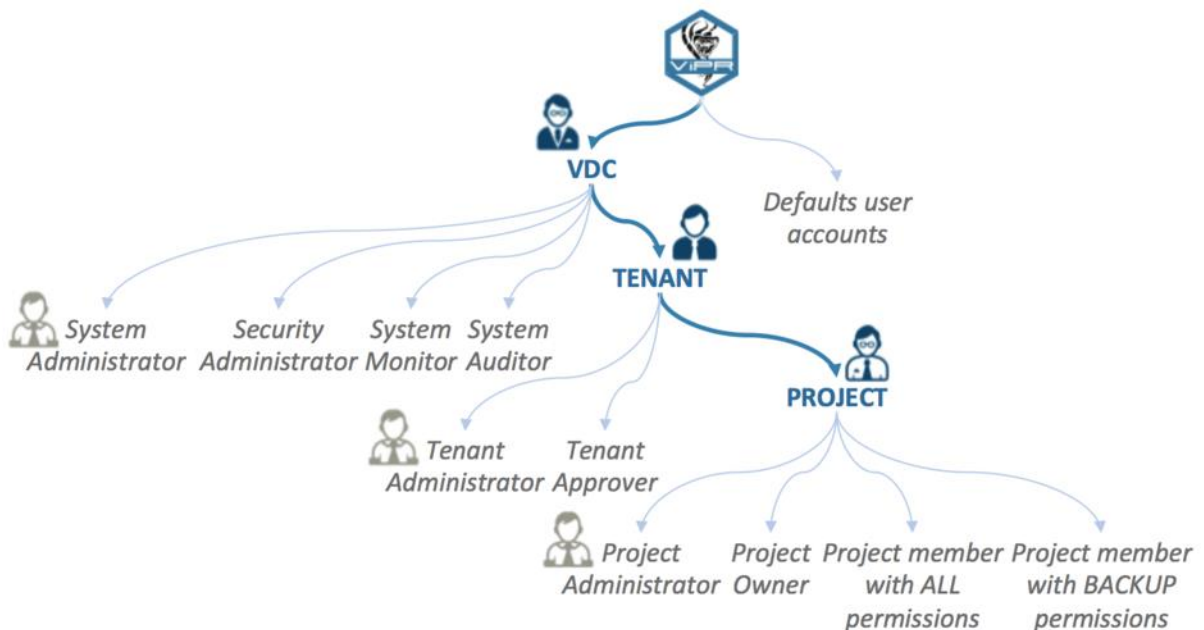






Figure 50: **Good** example of a diagram esthetic; Readers more easily perceive and remember the information.

To make less boring, straightforward procedures in the Simple  context, we can decorate corresponding diagrams. Complicated  and Complex  domains should be clean in respect to unnecessary decorations to decrease the overall level of informational noise. In the Chaotic  domain, presence or absence of decorations does not matter.

Use colors right

(Ware, 2013) and (Cairo, 2012) offer good advice on how to use color to encode information in diagrams:

- Use more **saturated** colors when color-coding small symbols, thin lines, or other small areas. Use less saturated colors for coding large areas.

- If using color saturation to encode numerical quantity, use greater saturation to represent greater numerical quantities. Avoid using a saturation sequence to encode more than three values.
- Consider using **red**, **green**, **yellow**, and **blue** to encode small symbols. Ensure luminance contrast with the background.
- To create a set of symbol colors that can be distinguished by most color-blind individuals, ensure variation in the **yellow–blue** direction. Avoid red-green pallet.
- Avoid using **gray scale** as a method for representing more than a few (two to four) numerical values.
- Pure colors are uncommon in nature. Therefore, to attract audience attention, use **softened hues** such as light blue or gray for everything else.

Limit your selection of colors to **six**. In some cases, up to 12 can be used but that is the absolute maximum. That principle allows selecting colors that are different enough that they can be differentiated easily.



Figure 51: Choose the colors from this list of 12 most efficiently used in diagramming.

When creating your color palette, keep in mind that the most useful color schemes based on the **color wheel** are:

- **Complementary** – two colors opposite each other on the color wheel. That scheme is suitable if you need contrast between different objects. However, text on complimentary background is too jarring to perceive.
- **Split-Complementary** – an adaptation of the complementary scheme. Two additional colors are adjacent to the base color. That scheme has high visual contrast but has less tension than in Complementary. It is quite popular in design diagramming.
- **Analogous** – colors that are next to each other on the color wheel. We see a lot of this harmonious scheme in nature. One color dominates, the second supports it, and the third is used to make an accent.
- **Rectangle** – uses four colors arranged into two complementary pairs.

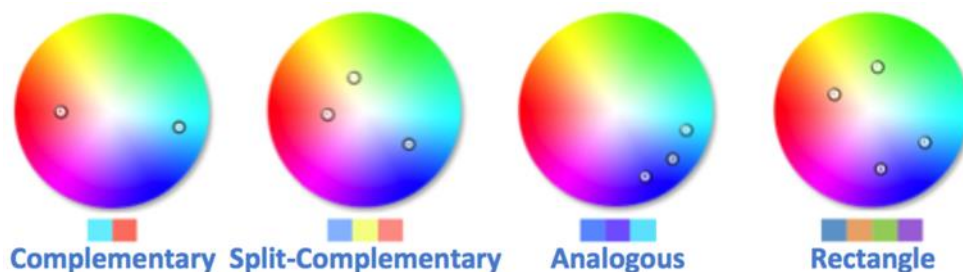


Figure 52: Color schemes helps to choose visually attractive palettes those can be used for information encoding;

Generated by means of sessions.edu/color-calculator

In the perspective of physics, colors are ordered by their wavelength. However, our brains do not naturally order colors. Still, brightness and saturation can be altered quite efficiently to represent the relative intensity of something (as in a heat map). In such diagrams, consider varying brightness along one or maximum two colors. For *example*, increasing the darkness of the blue can represent utilized capacity. Use of several **rainbow** colors is not a good idea.

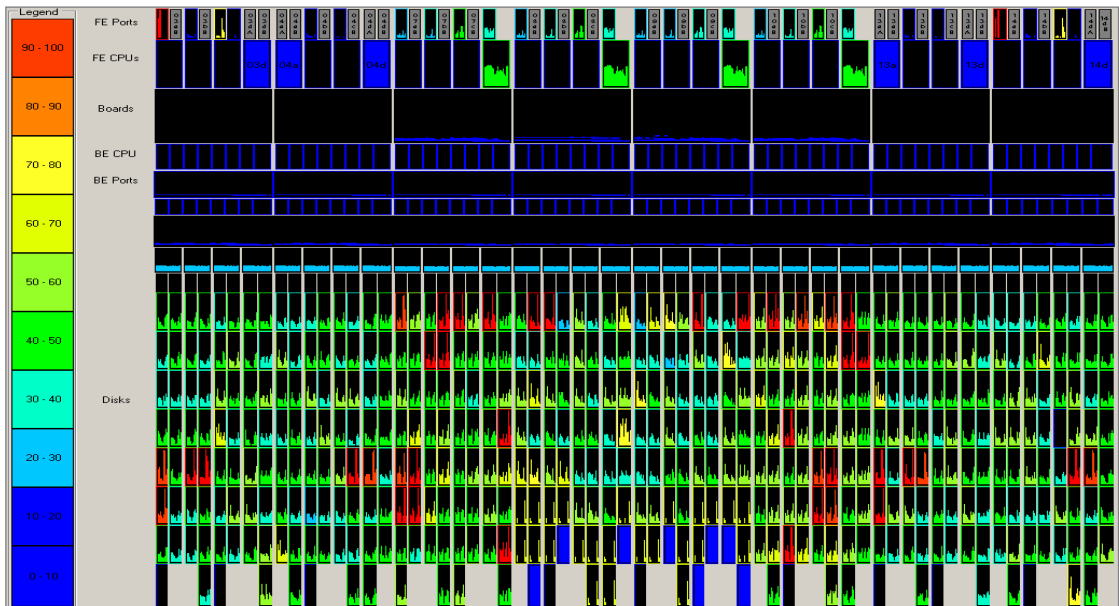


Figure 53: **Bad** example of intensity representation; Rainbow colors and the black background are used on the heat map. Bright green and blue unintentionally attract too much attention. Diagram is still informative because of tiny charts. Those are used as the redundant dimension to encode the data. Legend placed very close to the diagram helps as well.

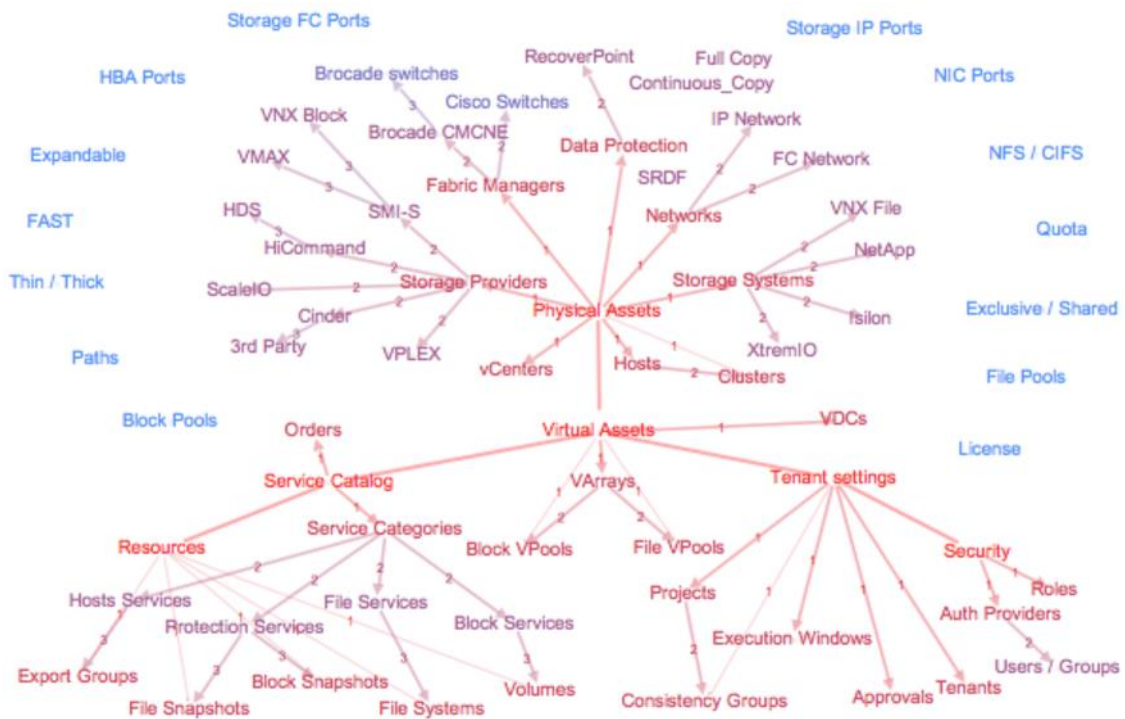


Figure 54: **Good** example of intensity coloring; Relative remoteness from the central nodes is represented by only two-color scale (red-blue). Redundant encoding with labels on edges and arrows on their ends supports that info.

Parameters with the maximum remoteness are not connected to other nodes to decrease visual noise.

Use the right fonts

Proper font helps to correctly perceive the diagram. As (Denneman) said in his blog: “Colors evoke feelings and moods, while the font determines the tone of voice in which the message is broadcasted”.

Common advice to use fonts are:

- For titles and labels it is better to use **Helvetica**, **Arial**, or **Verdana** fonts (**sans serif** fonts with clean line endings). Author: I like Calibri.
- For blocks of text it is better to use **Times**, **Cambria**, or **Garamond** fonts (**serif** fonts with small lines at the end of the strokes of a letter).
- Avoid gothic, fantasy, and script fonts such as popular Comic Sans.
- Use bold, italic, and underline font weights to emphasize.
- Use ALL CAPS text if you like to highlight or make an accent on something.
- Dark blue font on a white background gives a rich feeling to the labels.
- Choose just one or two fonts and be consistent with that for the entire set of diagrams.

Of course, this advice is not dogma. However, they are helpful if you do not feel confident with typography.

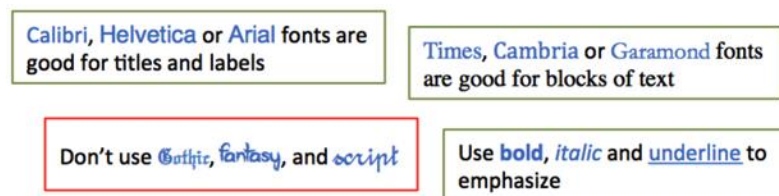


Figure 55: Right and wrong fonts.

Use paper, whiteboard, and everything you want

Most of diagrams presented in this article were created by the author on his computer. Such diagrams look more tidy and attractive than those drawn by hand. However, first drafts of many of the presented diagrams were sketched in the author's scratch pad. Paper is still quite a convenient media for a creative quick and dirty prototyping.

Please don't forget that flipcharts, whiteboards, napkins, or even the ground as a perfect “canvas” for a good solutions design.

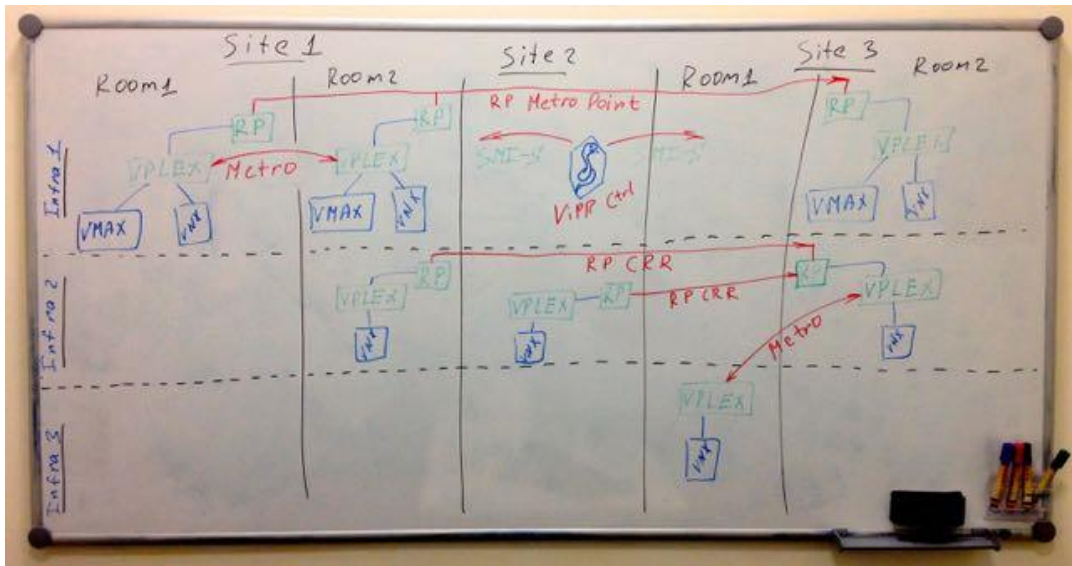
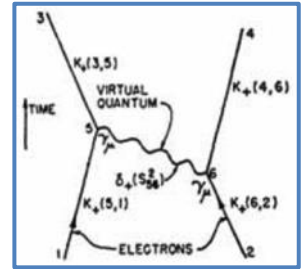


Figure 56: Solutions designers often use whiteboard and flipcharts; A whiteboard is shown with the diagram of complex physical infrastructure prepared for the ViPR implementation design.

Section summary

- Redundant encoding helps better perceive information.
- Incompatibility with the reality requires extra effort to decode diagrams.
- Structure helps to keep the diagram organized and simplifies pattern recognition.
- Readers easily recognize symmetrical visual patterns.
- Loose diagrams are easier to modify. Compact diagrams are more convenient to place and print.
- Noise reduces clarity. Decrease it when appropriate.
- Functional and emotional components of diagrams must be balanced.
- Limit the number of used colors and fonts.
- Make diagrams on any media you like.



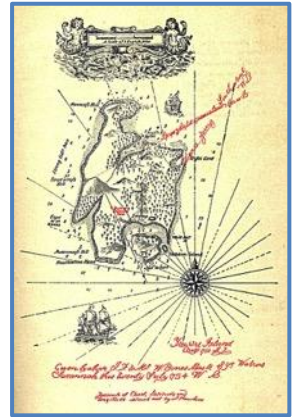
Feynman diagram; Richard Feynman, 1949

5 Conclusion

This article discussed pragmatic ways to enhance your design diagramming skills. Fundamental principles explain the diagramming philosophy. They clarify how efficient eXploratory and eXplanatory diagrams should be created and why these methods work. Pro tips, discussed in the article, help to accelerate and improve the process of diagram development in different contexts.

I have illustrated the article by pictures of ViPR solutions because they are related to my current work. As most of the illustrations were adapted from actual projects, the described approaches and tricks are not pure theory; rather, they are proven by the experience of successes and failures. Of course, all described principles are applicable to design of any other solutions.

The key point in diagramming is to understand why you need to create a diagram. The Idea is always primary. That Idea transforms the science of diagramming into the art of creative design.



Treasure Island Map, Robert Louis Stevenson, 1883

6 Bibliography

- *Basic color schemes - Introduction to Color Theory.* (n.d.). Retrieved from tigercolor.com: www.tigercolor.com/color-lab/color-theory/color-theory-intro.htm
- Bettencourt, L. (2010). *Service Innovation: How to Go from Customer Needs to Breakthrough Services.* McGraw-Hill.
- Cairo, A. (2012). *The Functional Art: An introduction to information graphics and visualization.* New Riders.
- *Causality.* (n.d.). Retrieved from Wiki: <http://en.wikipedia.org/wiki/Causality>
- Christianson, S. (2012). *100 Diagrams That Changed the World: From the Earliest Cave Paintings to the Innovation of the iPod.* Plume.
- Cleveland, W., & McGill, R. (1984). Graphical Perception: Theory, Experimentation, and Application to the Development of Graphical Methods. *Journal of the American Statistical Association* , 79.
- *Color wheel.* (n.d.). Retrieved from Wiki: en.wikipedia.org/wiki/Color_wheel
- *Correlation and dependence .* (n.d.). Retrieved from Wiki: en.wikipedia.org/wiki/Correlation_and_dependence
- Craig, M. (2013). *Communicating with Diagrams.* CreateSpace Independent Publishing Platform.
- Denneman, F. (n.d.). *10 guidelines for creating good looking diagrams.* Retrieved from frankdenneman.nl: frankdenneman.nl/2013/02/01/10-guidelines-for-creating-good-looking-diagrams/
- *Diagram.* (n.d.). Retrieved from [vocabulary.com](http://www.vocabulary.com/dictionary/diagram): <http://www.vocabulary.com/dictionary/diagram>
- Gothelf, J. (2013). *Lean UX: Applying Lean Principles to Improve User Experience.* O'Reilly Media.
- Hasan , H., & Kazlauskas, A. (2009). Making Sense of IS with the Cynefin Framework. *Pacific Asia Conference on Information Systems (PACIS).* Association for Information Systems.
- Higgins, J. (1994). *101 Creative Problem Solving Techniques .* New Management Publishing Company.

- Iliinsky , N., & Steele, J. (2011). *Designing Data Visualizations: Representing Informational Relationships*. O'Reilly.
- Kekkonen, K. (2012). *User-Centric New Service Development in Telecom Industry*. Aalto University School of Science .
- Kulpa , Z. (1994). Diagrammatic representation and reasoning. *Machine GRAPHICS & VISION*, 3, pp. 77-103.
- Kurtz, C., & Snowden, D. (2003). The new dynamics of strategy: Sense-making in a complex and complicated world. *BM SYSTEMS JOURNAL*, VOL 42, NO 3 .
- Lidwell, W., Holden, K., & Butler, J. (2010). *Universal Principles of Design, Revised and Updated: 125 Ways to Enhance Usability, Influence Perception, Increase Appeal, Make Better Design Decisions, and Teach through Design*. Rockport Publishers.
- Nakatsu, R. T. (2009). *Diagrammatic Reasoning in AI*. Wiley.
- Pantyukhin, V. (2012). *Infrastructure assessments cookbook: ingredients, technics and secret sauces*. EMC Proven Professional Knowledge Sharing.
- *Principles of grouping*. (n.d.). Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Principles_of_grouping
- Reynolds, G. (2007). *Signal-to-Noise ratio and the elimination of the nonessential*. Retrieved from presentationzen.com: http://www.presentationzen.com/presentationzen/2007/03/signaltonoise_r.html
- Srinivasan, K. (2012). *Blended Service Design Approach in Designing A Wiki Portal for Open Collaboration*. Leppävaara: Laurea University of Applied Sciences.
- Stickdorn, M., & Schneider, J. (2012). *This is Service Design Thinking: Basics-Tools-Cases*. BIS Publishers.
- *The power of visual grouping*. (n.d.). Retrieved from The elearning coach: <http://theelearningcoach.com/media/graphics/the-power-of-visual-grouping>
- Ware, C. (2013). *Information Visualization: Perception for Design, 3rd Edition*. Morgan Kaufmann Publishers.

EMC believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS PROVIDED "AS IS." EMC CORPORATION MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND WITH RESPECT TO THE INFORMATION IN THIS PUBLICATION, AND SPECIFICALLY DISCLAIMS IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Use, copying, and distribution of any EMC software described in this publication requires an applicable software license.