

Next Generation OSS

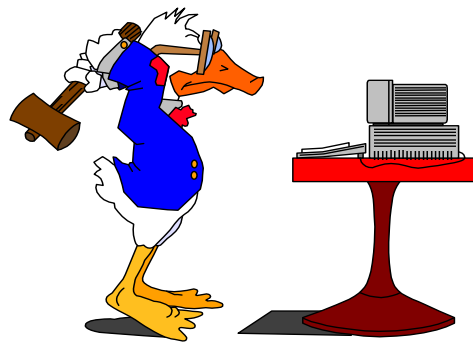
The Impact of Dark Fiber

- Because we know have control of the complete environment and have a more complex network we need a more considered approach to Network Operations
- NGOSS is the next big step forward in a Telecom world with cheap bandwidth and a scarcity of specialist personnel
- We in the Academic environment now have the opportunity to be involved with and contribute to the developments in this exciting area
- Starting from green field we can strive for an exceptional solution from the beginning
- I have been watching the NGOSS developments for some time and it seems that now they soon they may have something that we can use in our NOC

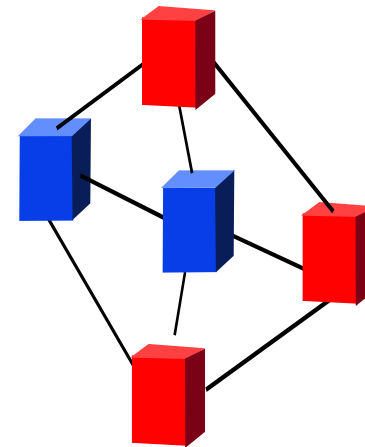


Time for a Radical Rethink

From 'blacksmith's shop'
software...



... to fast production line
by assembly and configuration

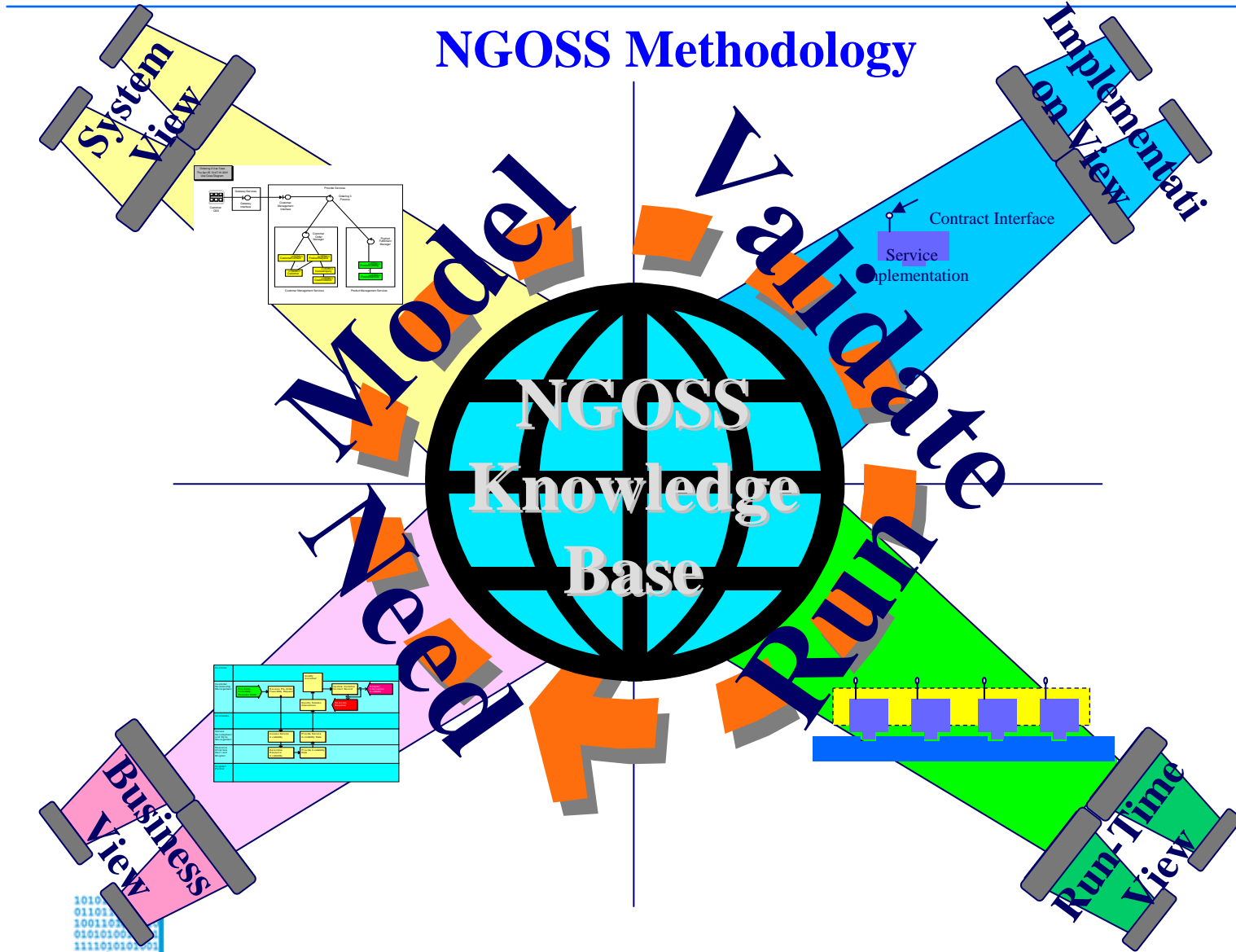


Next Generation OSS -
*changing the way we build
management software*

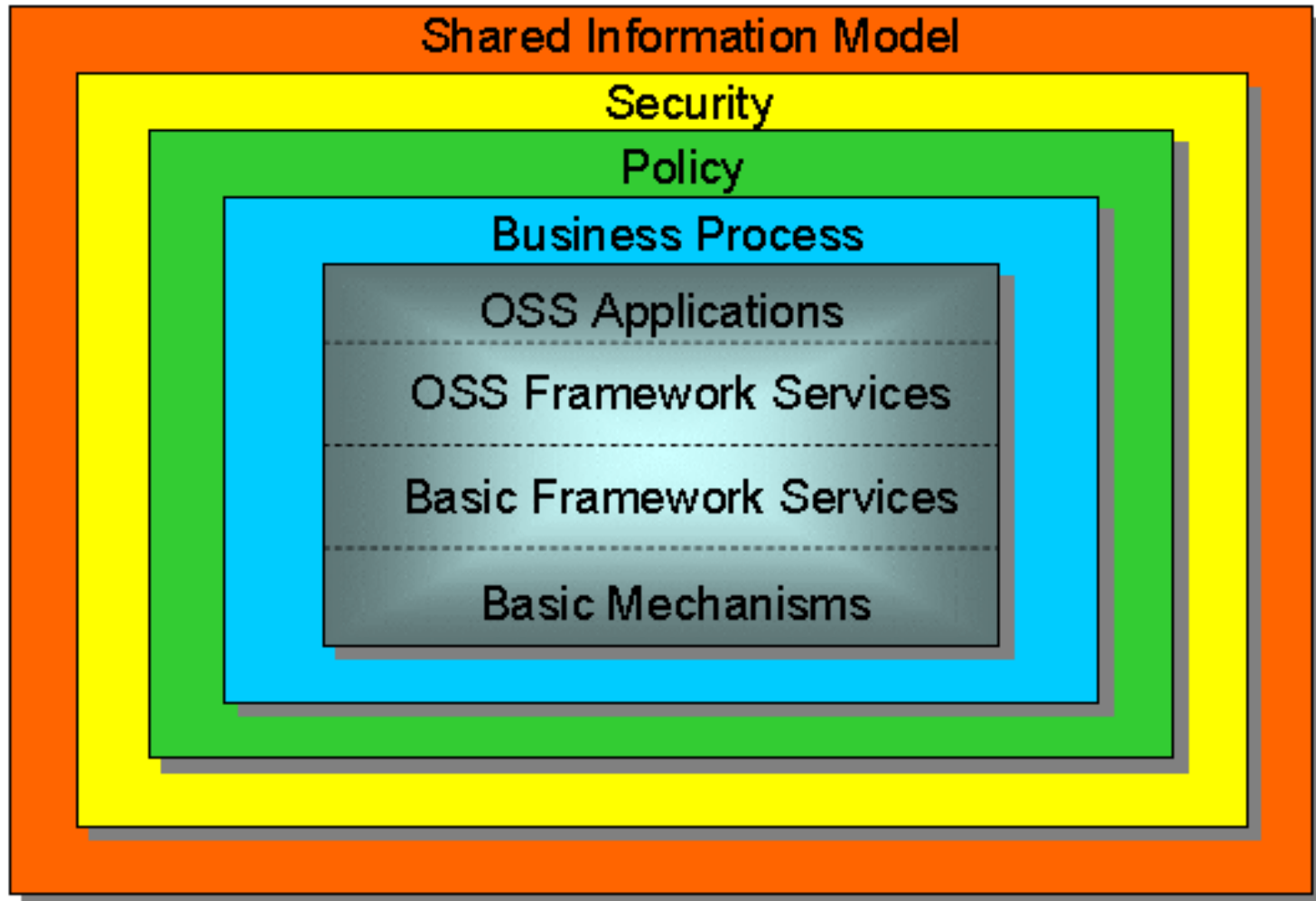


NORUnet

NGOSS Methodology



Technology Neutral Architecture



```
1111010101001
11010 0101010
00000 101010
01100 01101
```

Shared Information Model:

- An NGOSS system is characterized by a common information model to enable easy interoperability.
- The information model is more than just a standard representation of data
- It defines for managed entities:
 - semantics
 - behaviour
 - interaction
- A standard representation using standard data types
- Used to describe NGOSS domain information:
 - orders
 - network services
 - configuration definitions.



Security Policy Model:

- An NGOSS system should be designed according to an overarching security model.
- An implementation of an NGOSS system will require a security mechanism and policies so as to operate the NGOSS system in a secure manner.
- Security Management standards provide a framework to meet the company security objectives.



Policy Management Model:

- The NGOSS system architecture is defined as a policy-enabled management architecture.
- The term policy-enabled is defined as a system that operates using policies to make present and future decisions.
- Policies provide rules that govern the behaviour within a system.
- Policy rules are defined to meet business and system objectives
- Policies may link one or more of the business, operational and implementation views of the NGOSS system to each other.



Business Process Model:

- An NGOSS system is characterized by the separation of the hard coded behaviour of components from the software that automates business processes across the Components
- NGOSS system should be composed of defined services that can be orchestrated using process management technologies.
- An NGOSS system is further characterized by externalized descriptions of behaviour expressed in a technology-neutral manner.
- The external description of a Service is separated from one or more specific implementations through a representation called a Contract.
- Contracts enable different aspects of the business and system views of an application to be integrated through the use of business and system models.
- A business process is represented by an executable meta-language description, which describes the flow of execution between Component instances.



Business Process Model:

- System behaviour is controlled through the use of different types of management paradigms.
- A business process model may invoke lower-level business process models.
- This means that a business process step (e.g., verifying that the network can support the provisioning of a desired service) may be made up of one or more lower-level interactions with different system runtime entities that provide the necessary services;
- Lower-level business process models used in this way must be able to provide one or more Contract instances to which the higher-level business process model can bind.
- This means that multi-level process management can be supported.
- Process management is the application of modern business management techniques to business processes.



NGOSS Integrated Applications:

- NGOSS Integrated Applications are software components designed from the start for deployment in an NGOSS environment.
- Applications designed this way are designed and constructed with the following attributes:
 - Functionality accessible through NGOSS Contractual Interfaces,
 - Externally visible data conforms to a standard data model that can be mapped onto the NGOSS Shared Information and Data Model (SID),
 - Business Process and Policy functionality are separated from the implementation of the Application and specified using a set of Business Process and Policy rules (see previous Business Process Model).
- In this manner, Business Process and Policy Management techniques are applied to the separate Business Process and Policy Definitions which in turn orchestrates the flow between components of solutions and applications.



Integrated Legacy Applications:

- Integrated Legacy Applications are software components that were developed outside the scope of an NGOSS architecture and that have subsequently been made available as NGOSS components either by:
 - Fully encapsulating them with an NGOSS conformant wrapper (i.e., a software wrapper that provides access to the functionality of the legacy application through NGOSS Contractual Interfaces and maps data made visible by those Contracts onto the common information model used by the NGOSS deployment into which the legacy applications is integrated),
 - Encapsulating specific functionality of the legacy application by developing adjunct software which provides a Contractual Interface and data mapping for that specific functional interface.



Framework Services:

- These services provide the functionality to architecture implementations.
- Interfaces to these services are provided through appropriate contract specifications and can be further split into two sub classes:
 - OSS Framework Services
 - Basic Framework Services



OSS Framework Services:

- These Services provide standard OSS/BSS capabilities whose functionality is common to many OSS/BSS services.
- OSS Framework Services will most likely be included in some manner in all but the most trivial OSS/BSS deployment part of the NGOSS architecture
- Examples of OSS Framework Services include: Logging, Tracing etc.



Basic Framework Services:

- These Services provide the capabilities needed to support the patterns of interaction between Components implementing Business Services.
- The Basic Framework Services may also be used by the other Services.
- For example, the Process Definition for Billing would actually be found by fetching it from the Repository.
- The Basic Framework Services are not optional. Each NGOSS deployment must include at least one instance of each of the Distribution and Transparency Services that comprise the Basic Framework Services.



Distribution and Transparency Services:

- The TNA Distribution and Transparency Services are fundamental to the construction, deployment and use of solutions in an NGOSS environment.
- An instance of each Service must be included in each NGOSS deployment.

Distribution and Transparency Services:

- D&T Framework Services include:
 - **Naming:** to shield prospective users from the complexities (and inherent incomprehensibility) of network addresses,
 - **Repositories** (or registry services): Store and provide information about available distributed services,
 - **Registration Services:** Provide for the administration of services including the addition, modification and removal of services from the Repository, and the ability to browse services previously added to the Repository,
 - **Service Location Services:** Facilitate “matchmaking” between potential clients and servers. These Services accept requests for a particular service from potential clients and match that request against registered providers for that service stored in the Repository.



Federation Model:

- There are generally two types of methods of federating systems services and data:
 - either by providing a communications protocol between services providing similar functions (Service Level Federation),
 - or by providing mechanisms by which the underlying data (both system and user) can be shared (Repository Level Federation).
- Although both types of federation are recognized by the NGOSS architecture, due to scalability and performance issues introduced with the proliferation of pair-wise Service Level Federations, Repository Level Federation is the (strongly) recommended approach.



Repository Level Federation:

- Repository Level Federation requires an agreement on a common data model between the federating entities.
- In addition, since federation is at the Repository level (through a specialized Repository service interface implementing export and import Repository interfaces), federation is achieved transparently to all other Distribution and Transparency Services.
- For example, a client can request location of a Service deployed and advertised in a completely different domain and enterprise using the identical interface (and without knowing where the Service actually is) that would be used to locate a local Service.



Basic Mechanisms:

- These Mechanisms provide the baseline functionality which must be implemented in any NGOSS deployment to provide for the inter-working between independently deployed Components and to support the “plug-and-work” paradigm at the most fundamental level.

Common Communication Mechanism:

- An NGOSS system must be characterized by the existence of a communication mechanism (e.g., a messaging bus) or some other form of common communication.
- All software entities will use one or more communication mechanisms to communicate with each other.
- Each communication mechanism offers one or more different transport mechanisms.
- There may be more than one such communication mechanism within a given system implementation, and these mechanisms may represent different technology-specific mappings.
- The common communication mechanism must provide transport mechanisms consistent with the basic interaction styles defined in the architecture and any security policies that a service provider has defined for his/her network.



Common Communication Mechanism:

- More importantly, the use of a common communications mechanism enables the standardization of system-wide operations, messages, or events that can be distributed to interested components.
- This is an important point, as a transport “just” carries information, and by itself doesn’t provide interoperability.



Invocation Mechanism:

- These mechanisms provide a common means of performing the steps associated with invoking an operation on a Service instance.
- There are several steps associated with invoking an operation on a service instance:
 - location of the instance providing the service,
 - proper usage of the communication mechanism,
 - results handling, etc.
 - Security and other policies must be applied at each step in the process.
- For this reason, Invocation mechanisms must be defined as a convenient way to perform all of these steps and ensure the integrity of the operation in the context of the overall system and the client invoking the service.



Questions?

Thank you & Questions please



NORDUnet