SES MDI using SDC/SDPi+FHIR
Framework & Considerations

Pre-release
Draft 2020.07.17

Establishing a framework for

Trusted Interoperable Product Decoupling

SES MDI Using SDC/SDPi+FHIR – *The Big Idea*

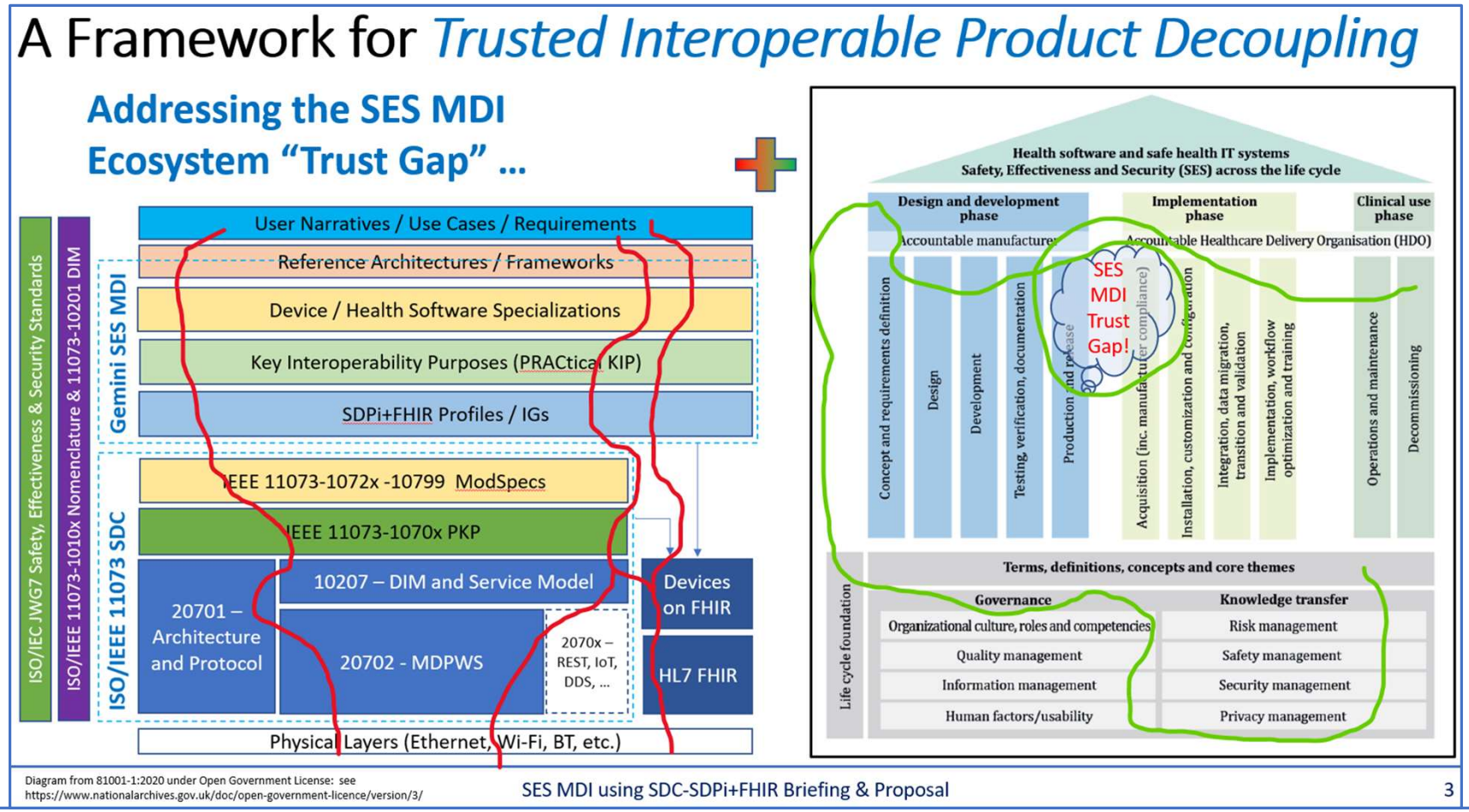


Diagram from 81001-1:2020 under Open Government License: see https://www.nationalarchives.gov.uk/doc/open-government-licence/version/3/

Full presentation available @ Gemini SDPi+FHIR "The SES MDI Interoperability Trust Gap" @ https://confluence.hl7.org/x/IQ7xB

Questions to Contemplate …

How to minimize duplication of information & effort?

How to maximize use of work and resources already created?

What is the requirements starting point & ending point?

How to truly integrate SES & MDI … seamlessly?!

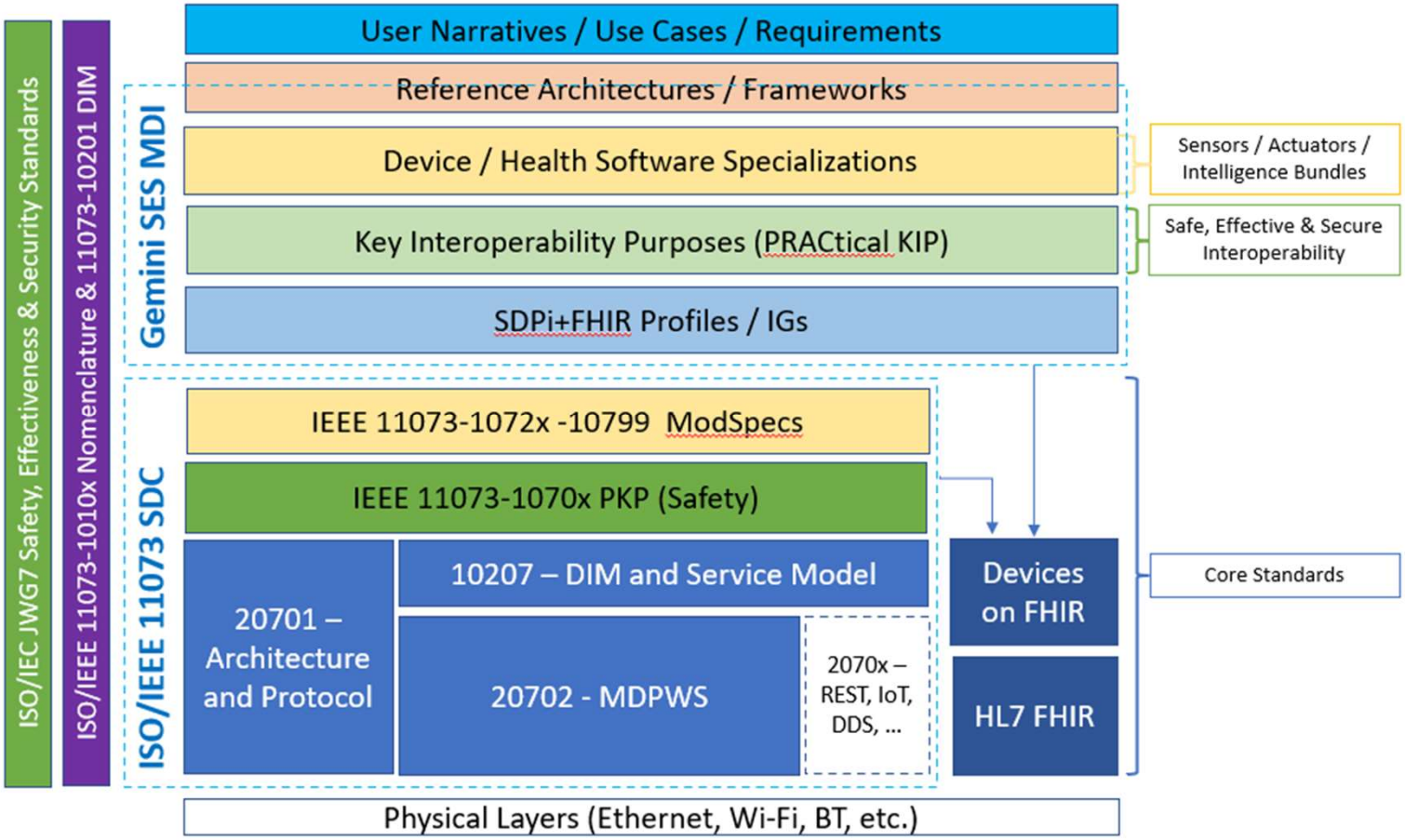
How to craft a framework for SES "PnT" MDI …

… With an achievable starting point & compelling future state?

SES MDI Using SDC/SDPi+FHIR – *The Big Picture*

"Hanging Gardens" Model

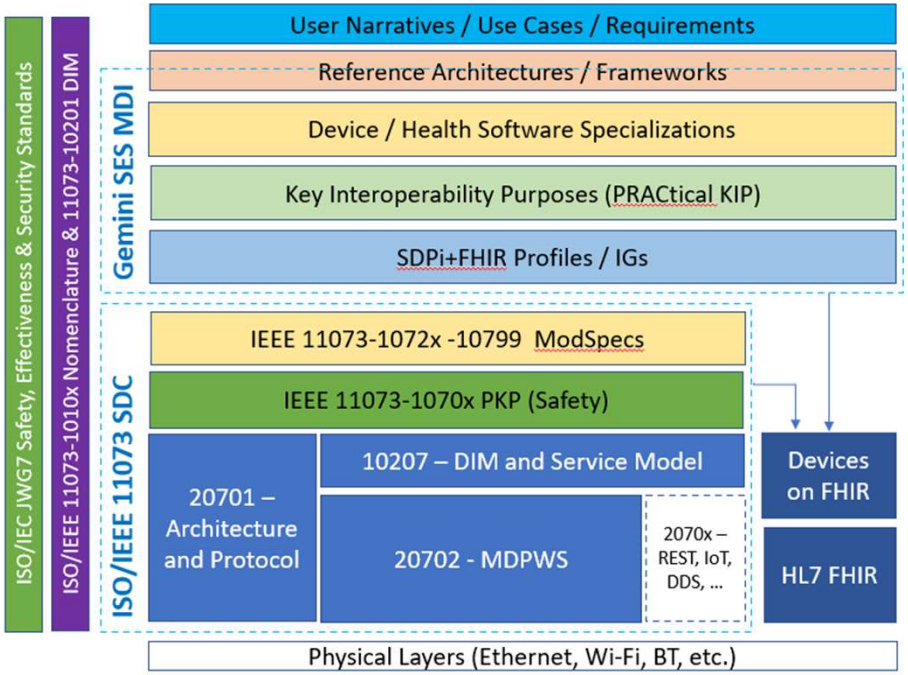
- ✓ Perfect? Not!
- ✓ Useful? Absolutely!!!
- ✓ Provides a common reference point to consider the questions and identify a workable framework
- ✓ Cuts across …
  - Standards
  - Organizations
  - Subject Areas
  - Past/Present/Future
  - ….



“Hanging Gardens” Model – *Layer Integration*

Each layer can be characterized by:

- Unique Subject Concepts & Components (Terms)
- Layer-specific Information & Knowledge (UML models)
- Layer API & Capabilities & Requirements (Inter-layer Spec’s)
- Requirements Formalization (Gherkin & ReqIF Spec’s)
- Implementation Trust Logic (SES Assurance Case Spec’s)
- Implementation Technology Logic (MDI Spec’s)



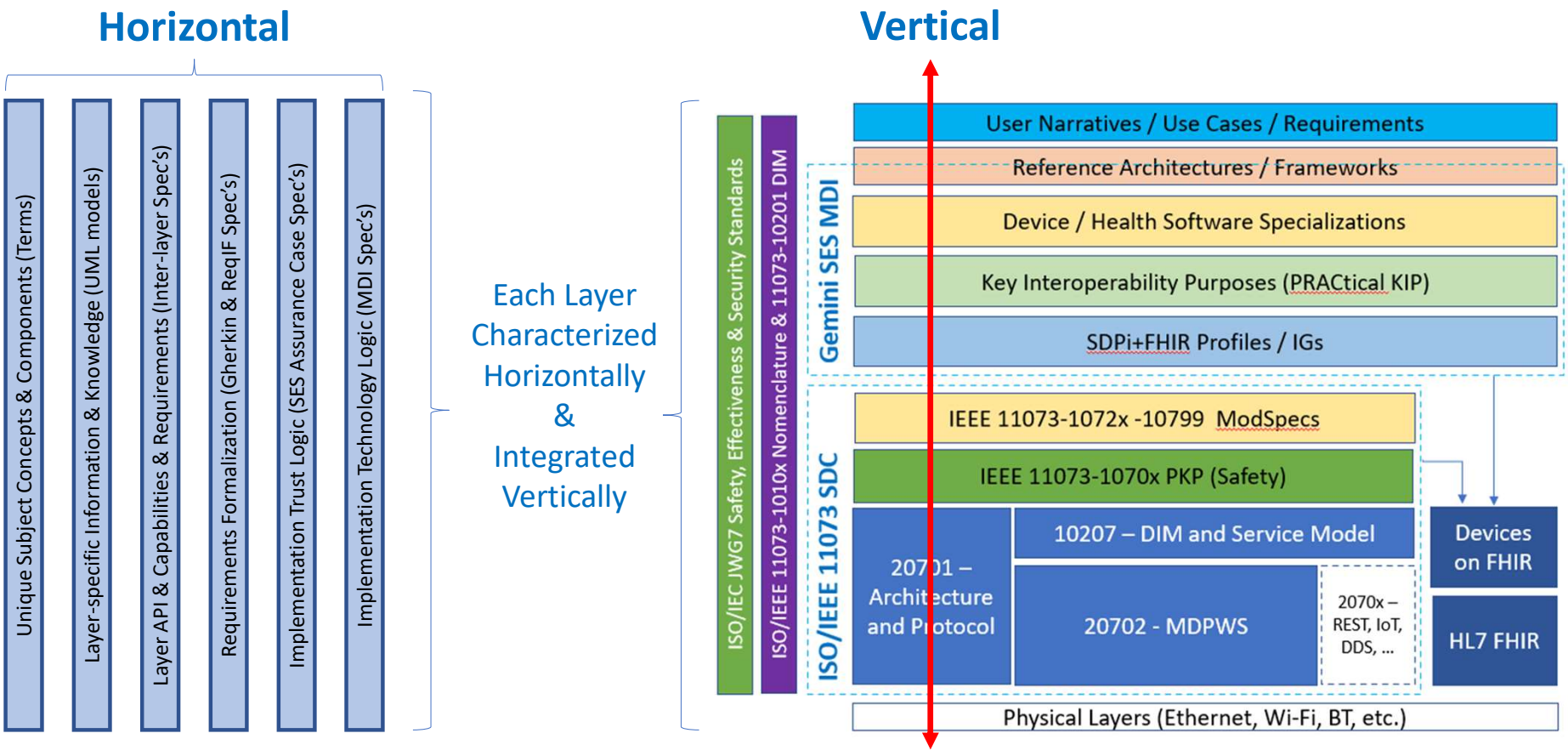
Integrating each “hanging garden” can be achieved by “threading” …

horizontally – across layer-specific characterization dimensions

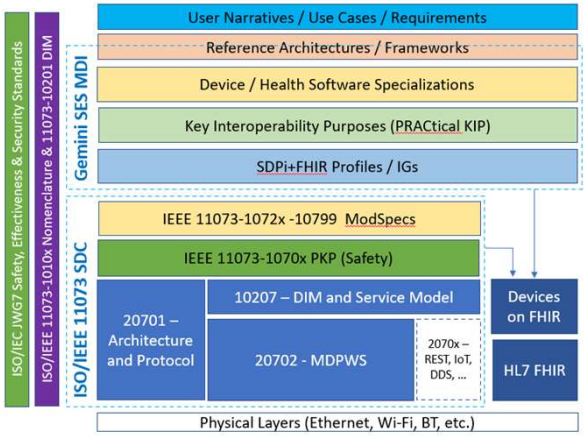
+

vertically – mapping from scenarios to Plug-and-Trust decoupled product interfaces

Layer Characterization: *Horizontal & Vertical*



Garden Design: *Horizontal Integration*



Unique Subject Concepts & Components (Semantics/Terms)

Concepts, terms, definitions, algorithms, frameworks, …

Layer-specific Information & Knowledge (UML models)

Use Case / Activity Diagrams, Sequence Models, Object Models, etc.

Layer API & Capabilities & Requirements (Inter-layer Spec's)

"API" Specifications, Implementation "Platform" Requirements, etc.

Requirements Formalization (Gherkin & ReqIF Spec's)

Feature, Rule, Given/When/Then + Specification, SpecRelation (Source, Target), etc.

Implementation Trust Logic (SES Assurance Case Spec's)

Profiled application of 81001-1, 80001-1, 62304, assurance cases, etc.

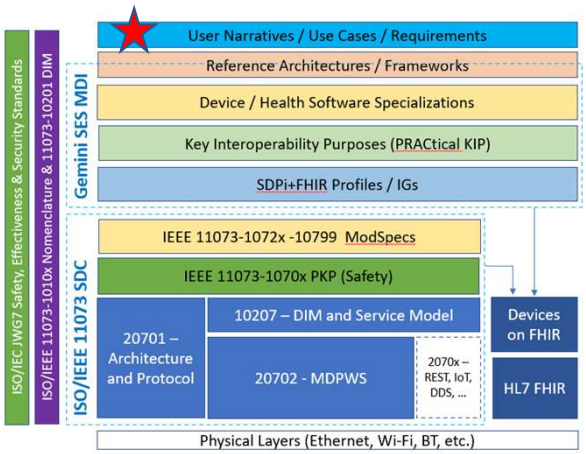
Implementation Technology Logic (MDI Spec's)

System Actors, Transactions, Messages, Events, Terminology, Value Sets, etc.

INCLUDE MAPPING COMPONENT / ARTIFACTS + standards in scope

Create a UML model showing relationships

Garden: *Use Case Scenarios*



Unique Subject Concepts & Components (Terms)

Narrative Stories, Actors (People & Systems), Scenarios, Activities, …

Layer-specific Information & Knowledge (UML models)

Use Case / Activity Diagrams, Sequence Models, Object Models, etc.

Layer API & Capabilities & Requirements (Inter-layer Spec's)

"API" Specifications, Implementation "Platform" Requirements, etc.

Requirements Formalization (Gherkin & ReqIF Spec's)

Feature, Rule, Given/When/Then + Specification, SpecRelation (Source, Target), etc.

Implementation Trust Logic (SES Assurance Case Spec's)

, etc.

Implementation Technology Logic (MDI Spec's)

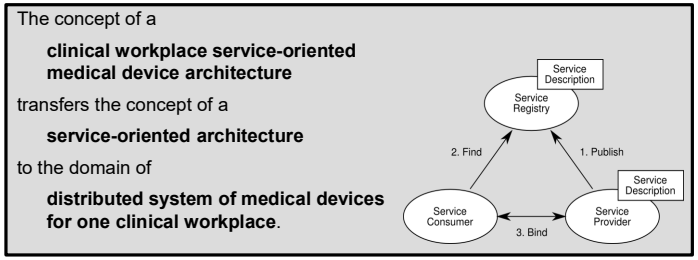
Actors, Transactions, Messages, Events, Terminology, Value Sets, etc.

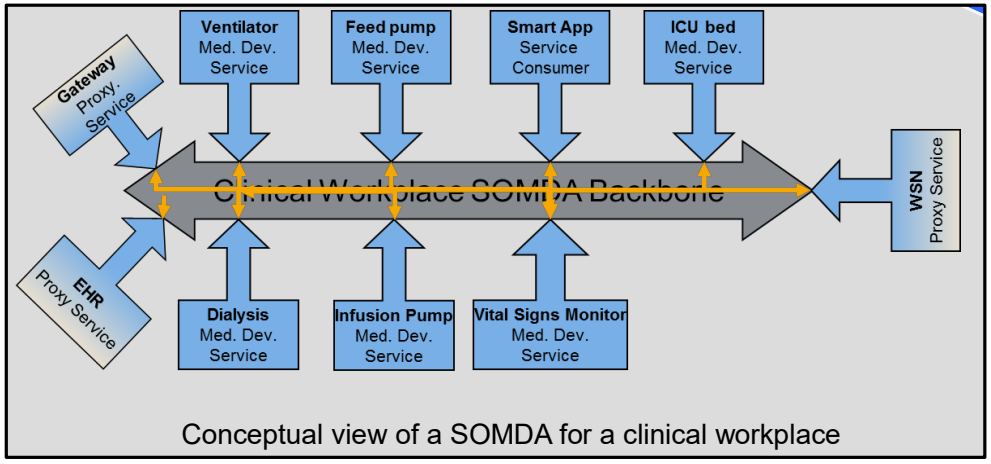
Update layout to focus on Use Case Stuff – Minimize horizontal / vertical

Add layer Scope / Garden Theme

Garden: *Reference Architectures*

- Architecture layer
- Include: MDIRA/ICE and SDC/SOA examples

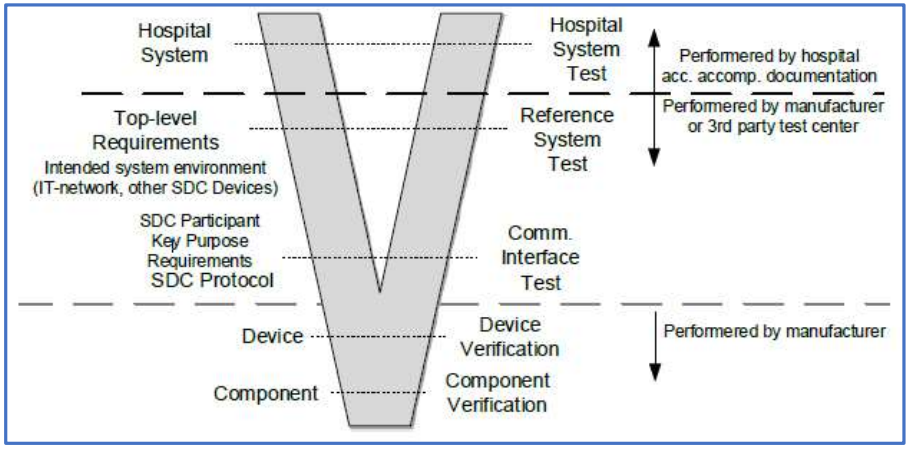




Conceptual view of a SOMDA for a clinical workplace

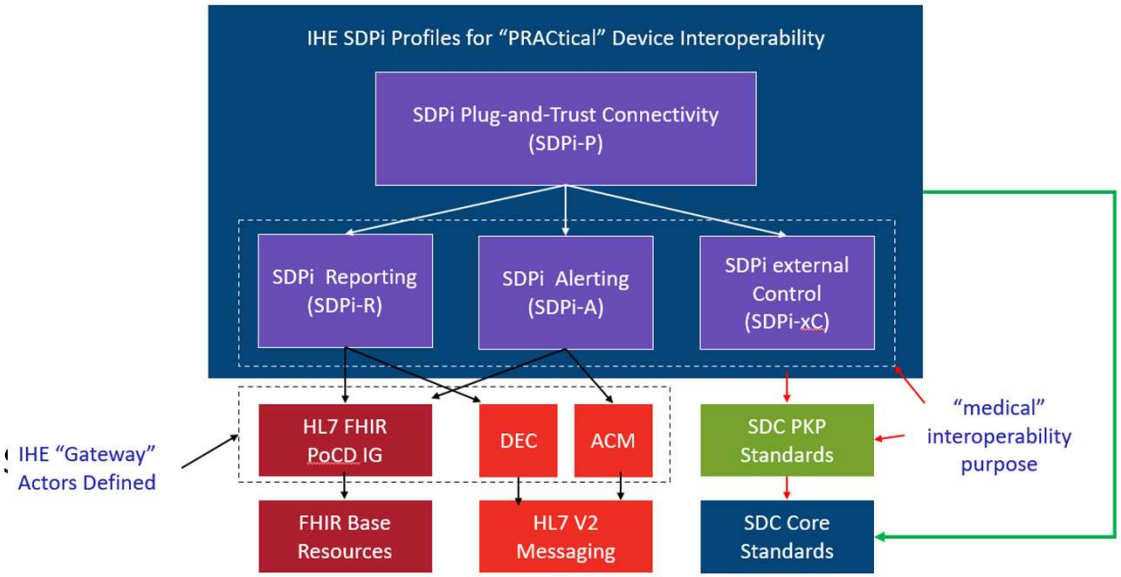
Garden: *Key Interoperability Purposes (SES)*

- SES KIP
- Includes:
  - SES profiles @ MDI standards
  - Assurance Case Templates
  - Interoperability model(s)
  - 4 Key Purpose Areas for MDI
  - Profiles for applying SES standards to SDC/SDPi+FHIR (incl. 81001-1, 80001-1. 82304, 62304, etc.)



Garden: *IHE SDPi*

- IHE SDPi
- IHE TF Constructs:
  - ✓ Use Cases (actors, transactions)



Garden: *ISO/IEEE 11073 SDC*

- SDC – ModSpecs, PKP, BICEPS, SOMDA, MDPWS
- <? 5 slides or one or … ???>
- <include mapping to TF volumes>
- Integrated Rxxxx
- BICEPS model
- SOMDA Model
- WS-*

Hanging Gardens: *Pulling it all together …*

- SES MDI (TF-1 Appendix A)
  - SFC – Specification / formalization / organization / …
  - Assurance Cases – composable / computable / V&V'able / CA'able
  - SES Requirements – per 81001-1, 80001-1, 60601-x-y, 62304, etc.
  - KIP(Regulatory) – sans implementation tech (see earlier slide)
  - CA @ SFC …

- ACID TEST:
  - ✓Requirement Test: Verify that PKP OID is in a device's security certificate
  - ✓Challenge: Do we know WHY it has to be there? Can we trace that requirement to the initial layer (Scenario?) that mandated its inclusion?

Additional Information

OMG Requirements Interchange Format (ReqIF) “Mapping” Background

OMG ReqIF: Base Model

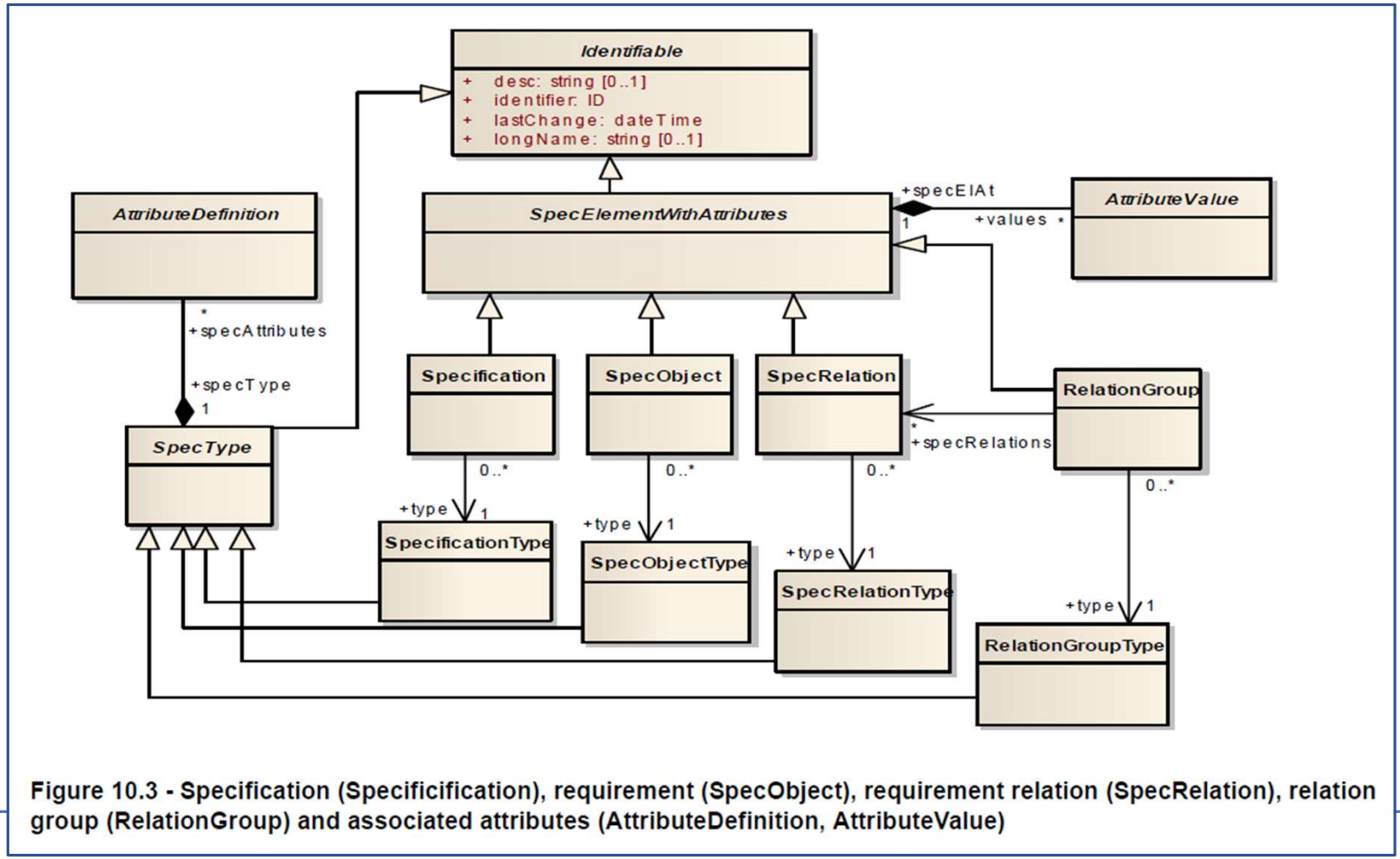


Figure 10.3 - Specification (Specificification), requirement (SpecObject), requirement relation (SpecRelation), relation group (RelationGroup) and associated attributes (AttributeDefinition, AttributeValue)

ReqIF: Requirement Hierarchies & Relationships

“mapping”
between
Source & Target Requirements

Two requirements may have a relation to each other, for example to establish traceability between a Customer Requirements Specification and a System Requirements Specification. Having a relation is represented by an association of one SpecRelation element to two SpecObject elements, one being the source, one the target of the relation.

The two specifications that are related to each other (in the above example: a Customer Requirements Specification and a System Requirements Specification) are referred to by the sourceSpecification and targetSpecification association of a RelationGroup instance.

The hierarchical structure of a requirement specification is represented by SpecHierarchy elements.

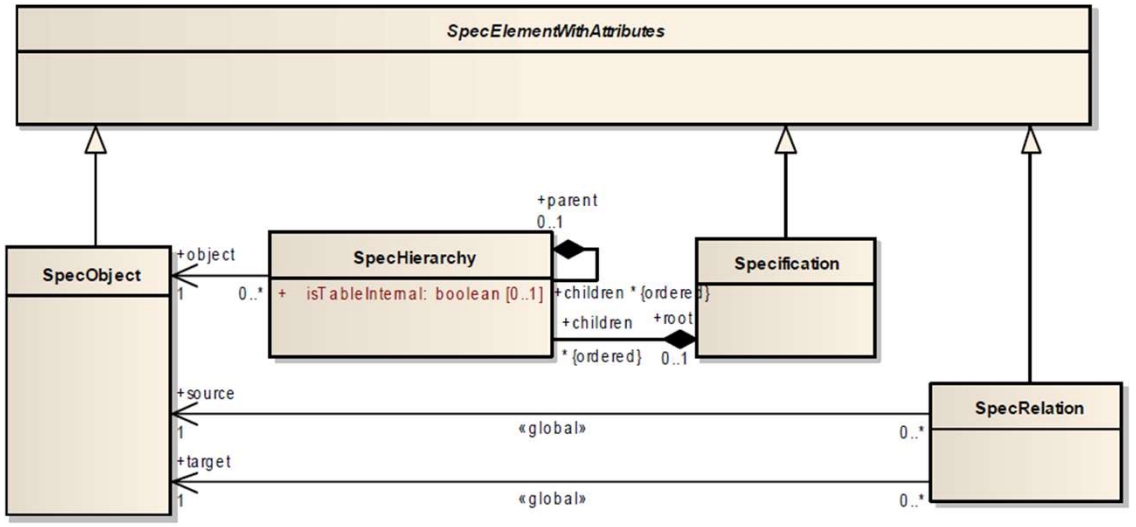


Figure 10.6 - Requirements, requirement relations and how requirements are structured hierarchically in a specification