

Model driven workflow

applied to an
IMS Application server

Antal Wu-Hen-Chang
antal.wu-hen-chang@ericsson.com
Tibor Csöndes
tibor.csondes@ericsson.com



Agenda



› Introduction

- Model-driven Design (MDD) and Model-based Testing (MBT) synergy
- IMS Application Server

› Model-driven Workflow

- Workflow overview
- Service implementation example

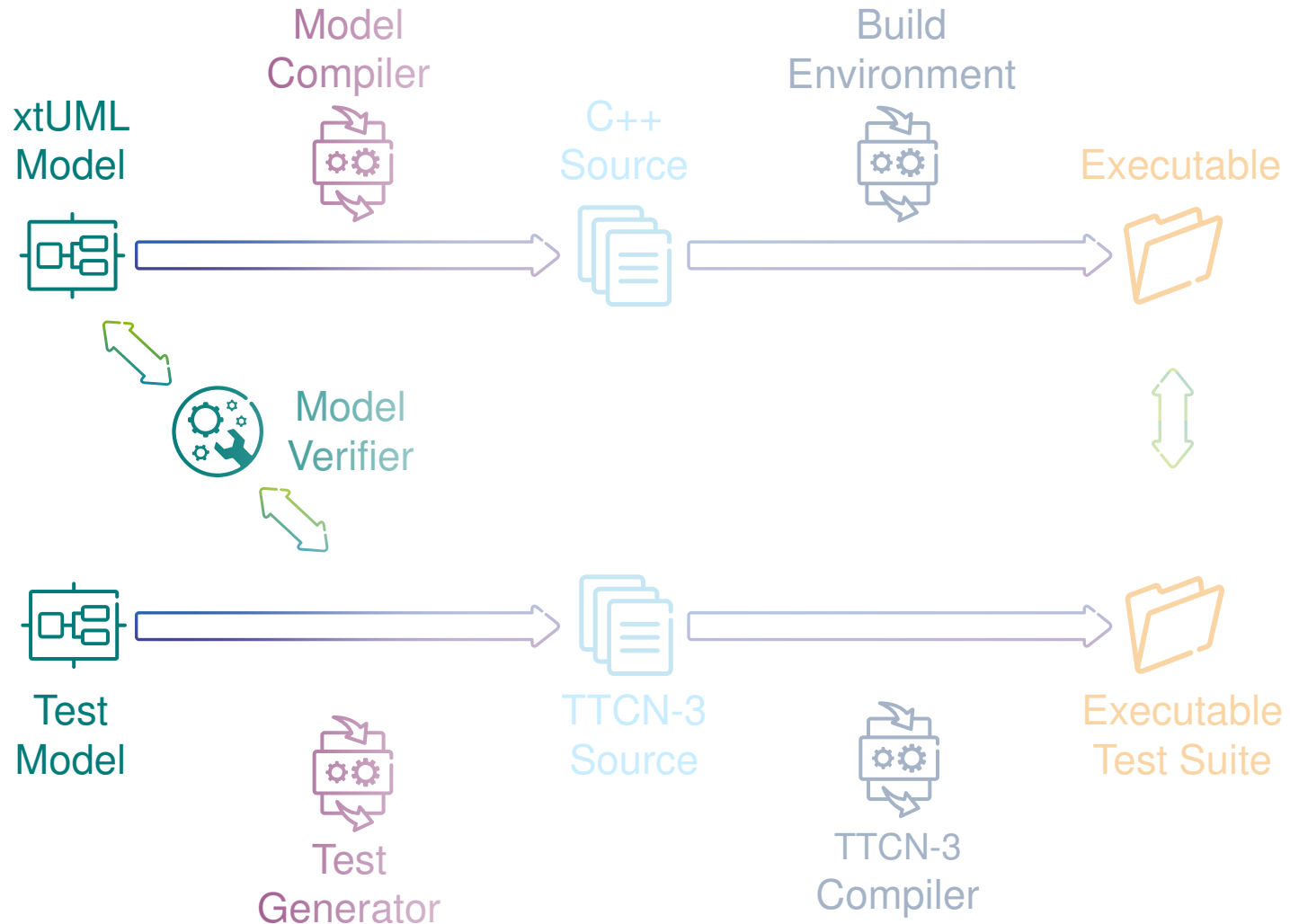
› Experiences

- Challenges and solutions
- Benefits

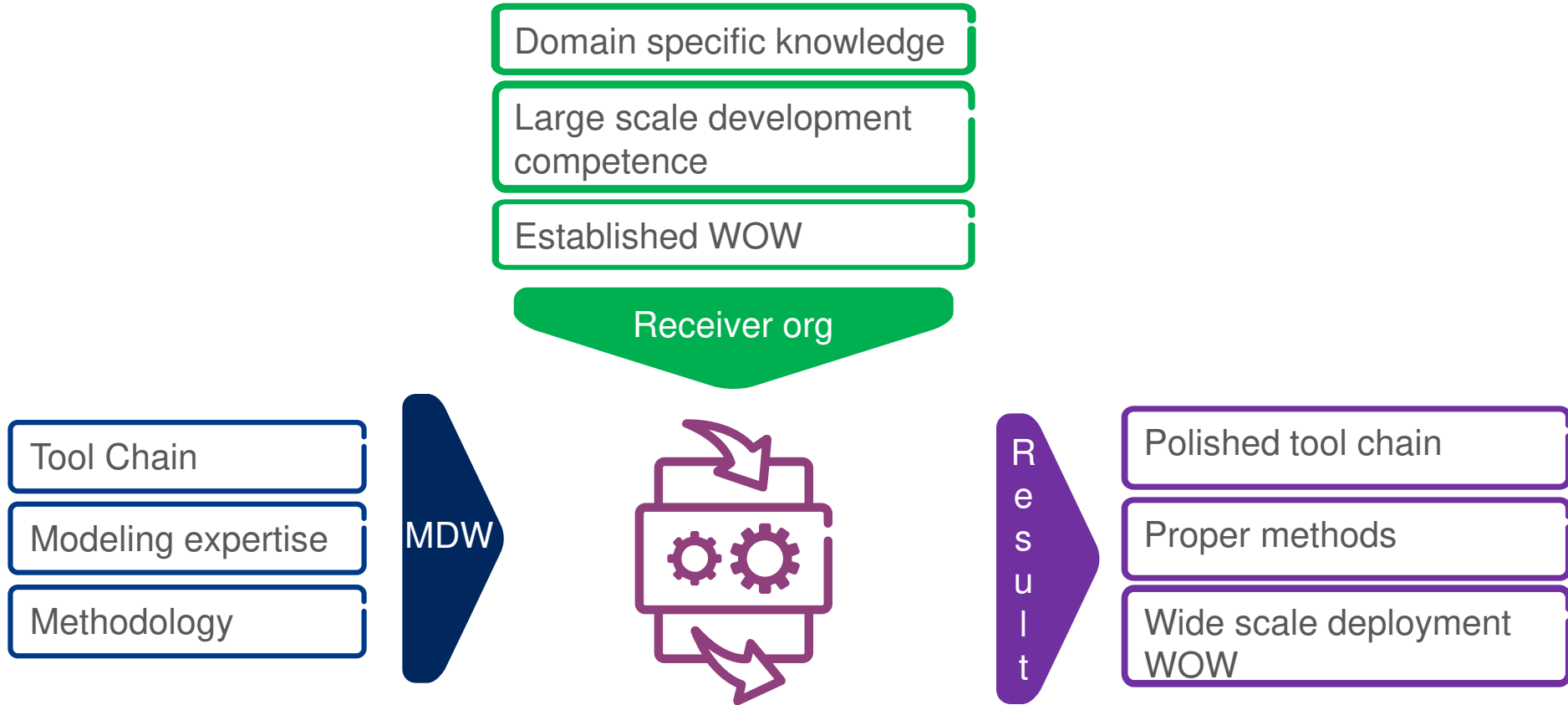


introduction

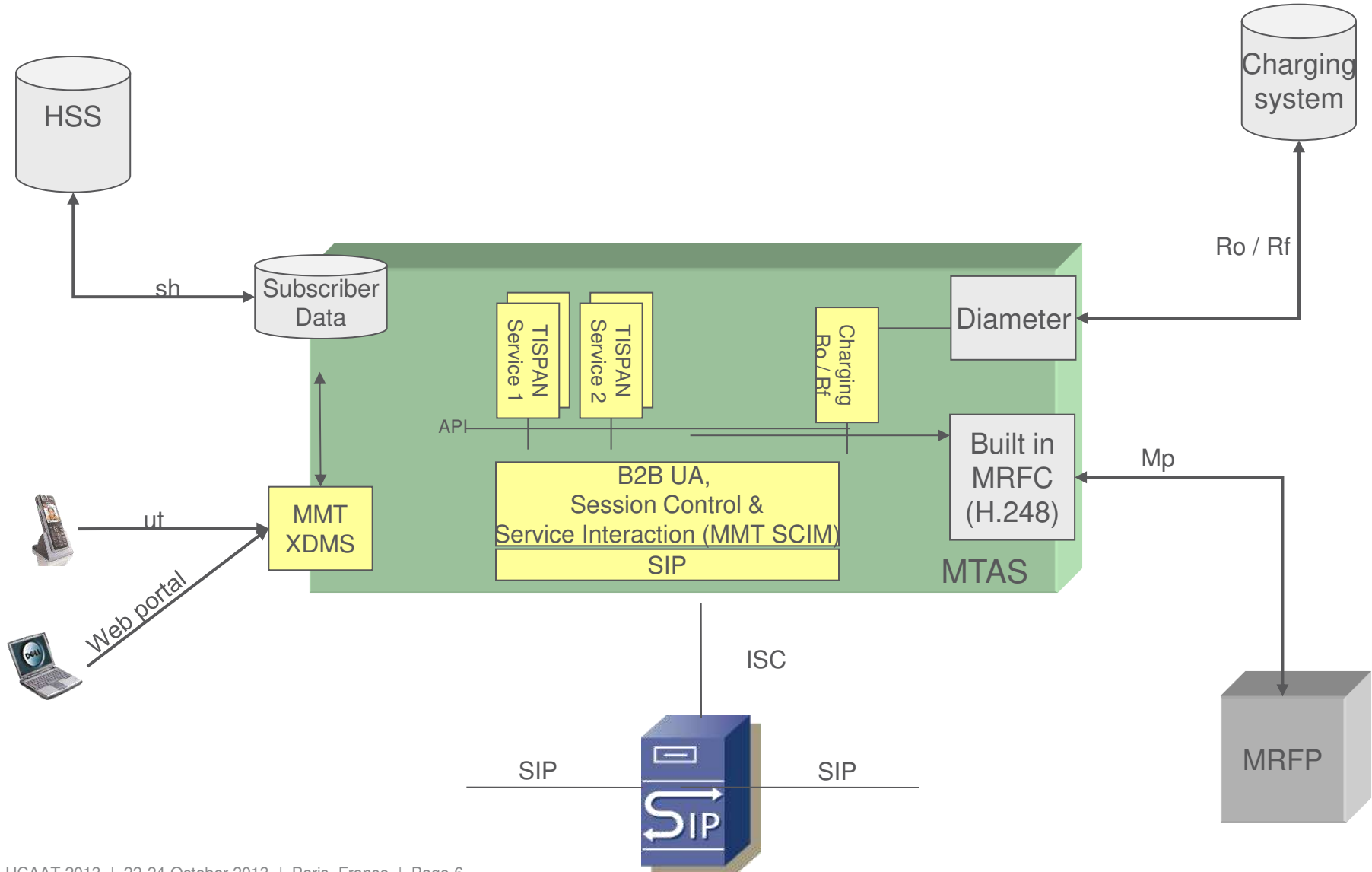
MDD and MBT synergy



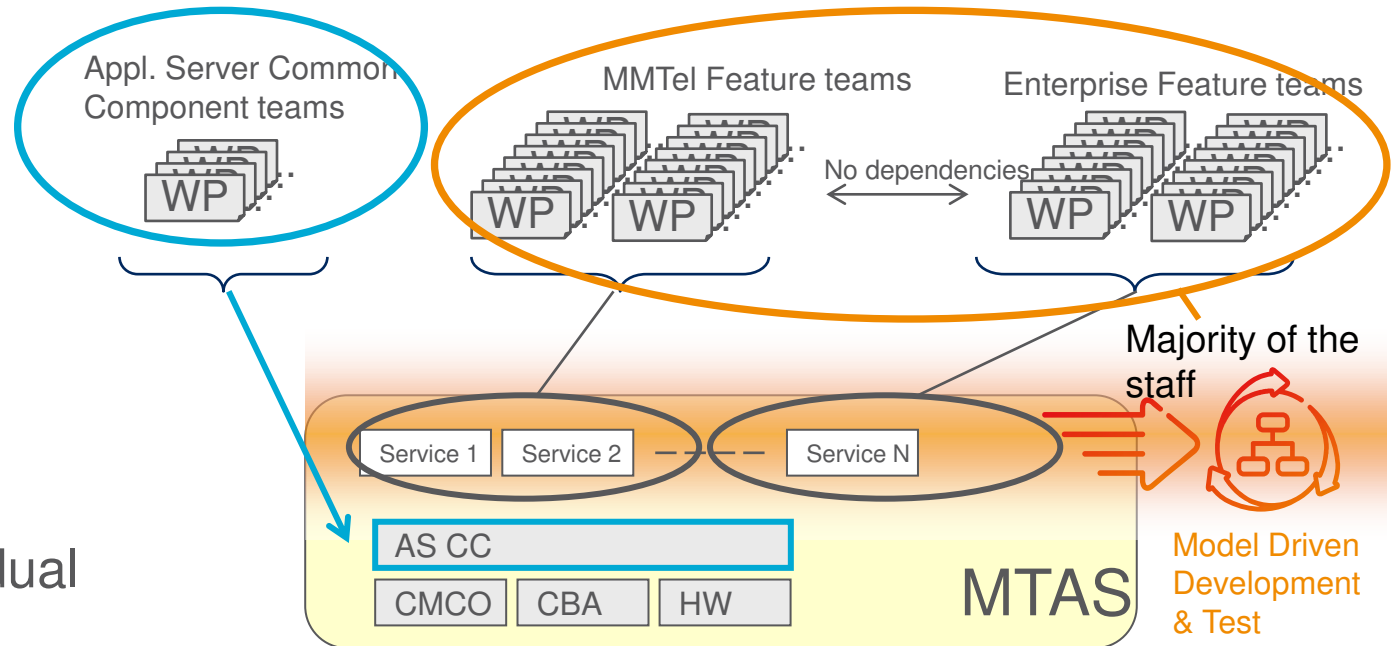
Mission



IMS application server



Workflow Introduction challenges



- Several teams
- Seamless gradual introduction
- Agile ways of working
- Huge handwritten legacy code
- Teams have no modeling knowledge

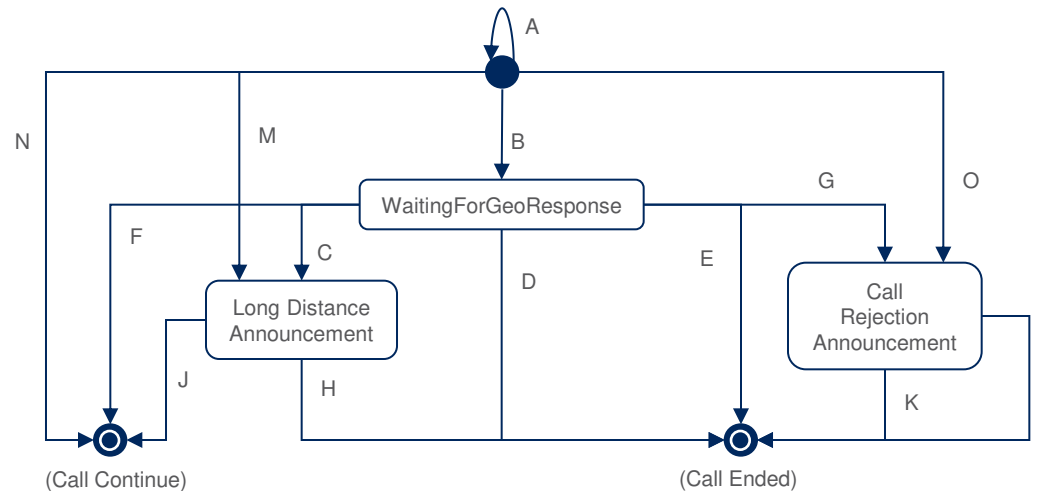


Model driven workflow

Service specification (original)



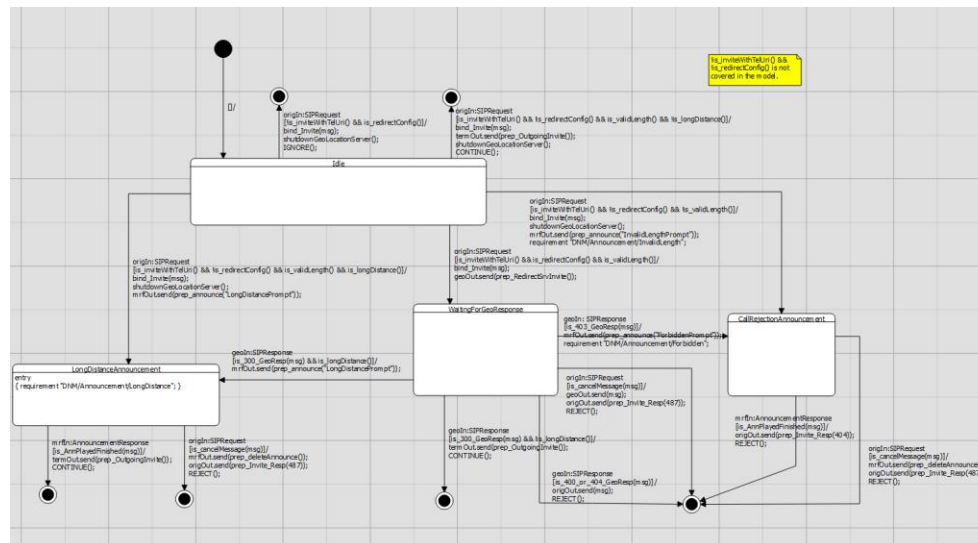
- New service requirement
- System architect designs the service logic in a natural language document
- Specification is given to the work package teams
- Parallel development of implementation and automated tests
- In the end of the sprints they can test their systems against each other



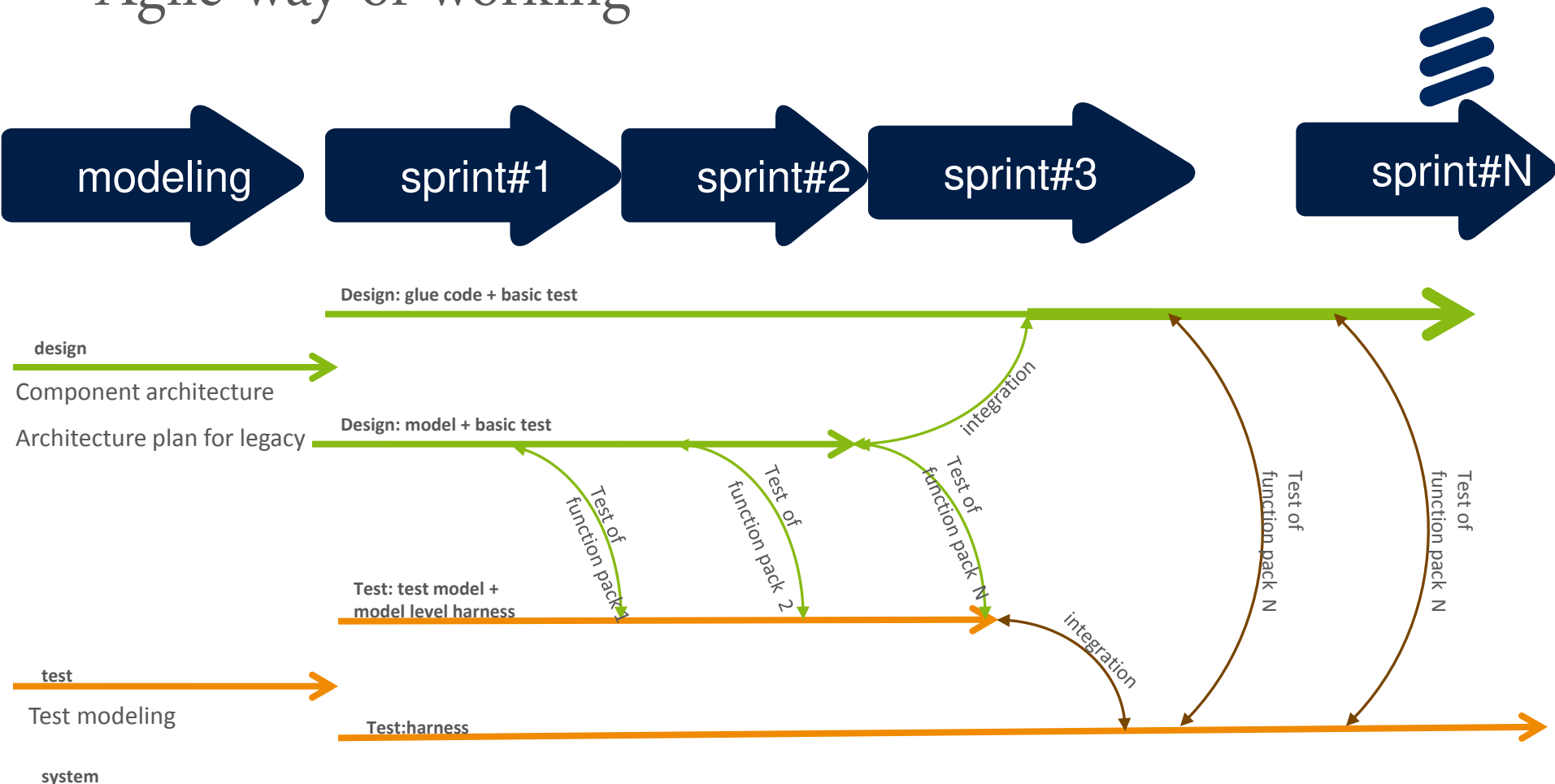
Service specification (modeled)



- New service requirement
- System architect designs the service logic in a **formal language**
- Specification is given to the work package teams
- Parallel development of implementation and automated tests
- **Early in the process** they can test their systems against each other



Agile way of working

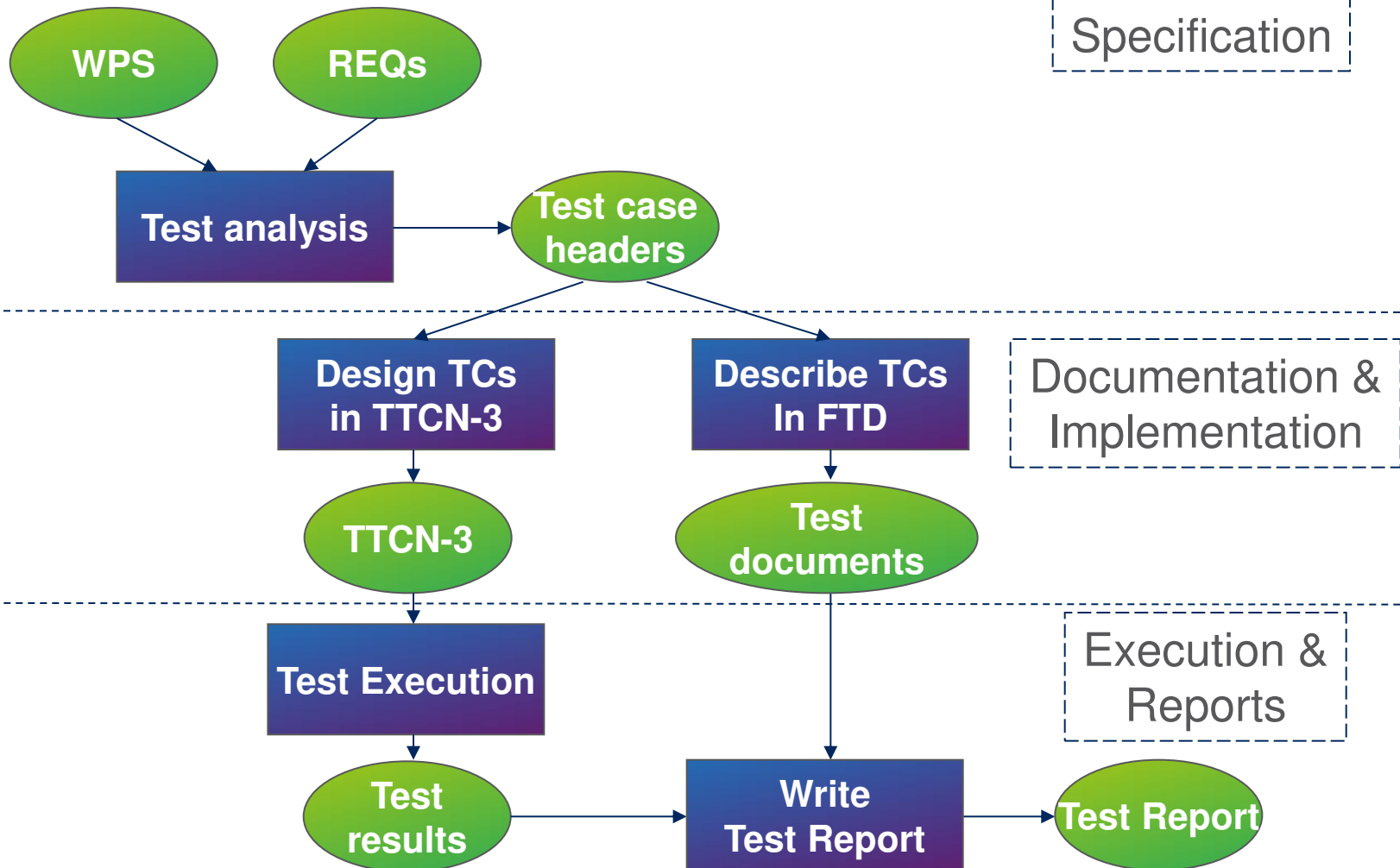


- Starting point: high level black box model
- Test driven development
- Continuous integration: nightly builds

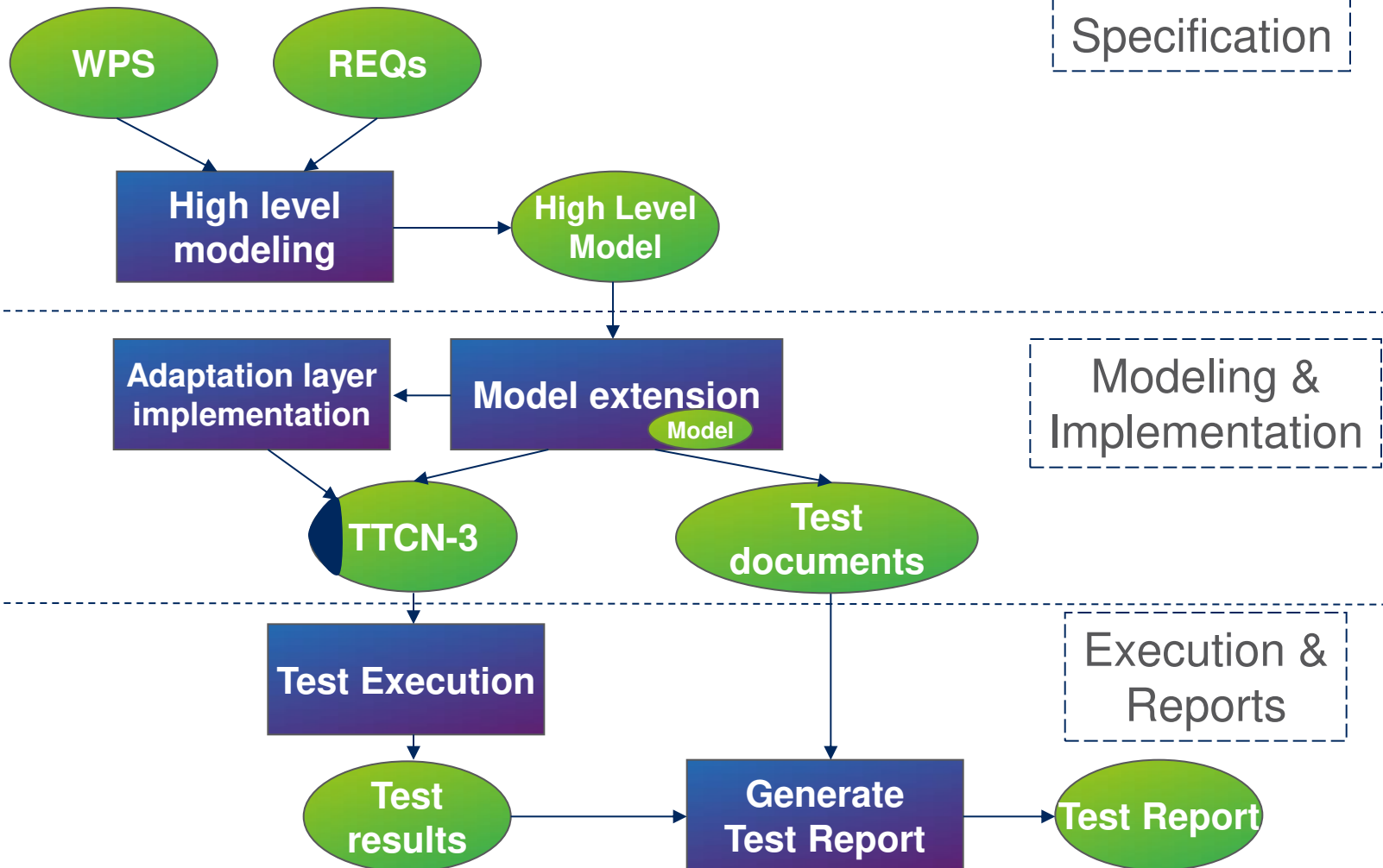


CSCF	MDW
3 designers	4 designers
3 testers	1 tester
1 system	2 MC

Original TEST workflow



Model based Test workflow

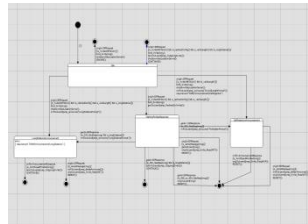


Model in the spec

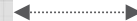
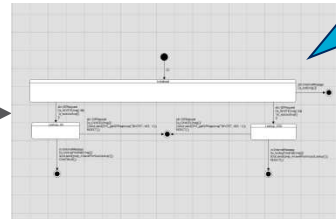


Service Logic

DNM Service Logic



Geo-Location Server



• Top down modeling
• Some of the signaling is pushed down

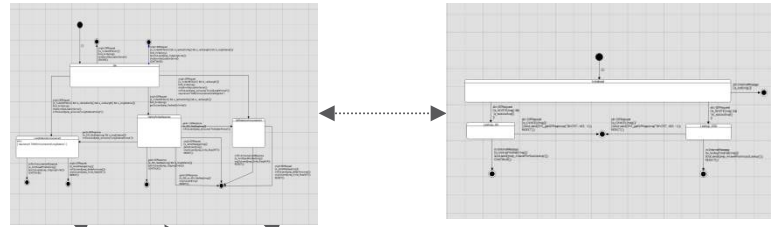
Model in the spec



Service Logic

DNM Service Logic

Geo-Location Server



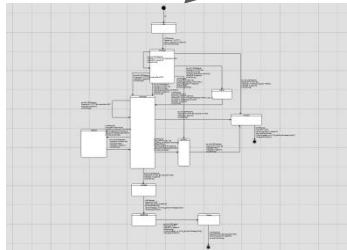
• Upper layer is evolved continuously

• No structural changes in the state machine

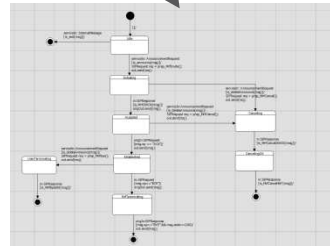
• Common parts are organized in a separate reusable libraries

Common Components

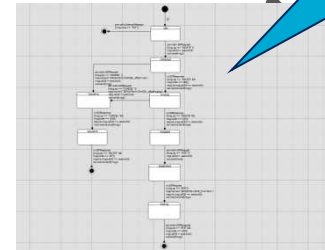
SIP Originating



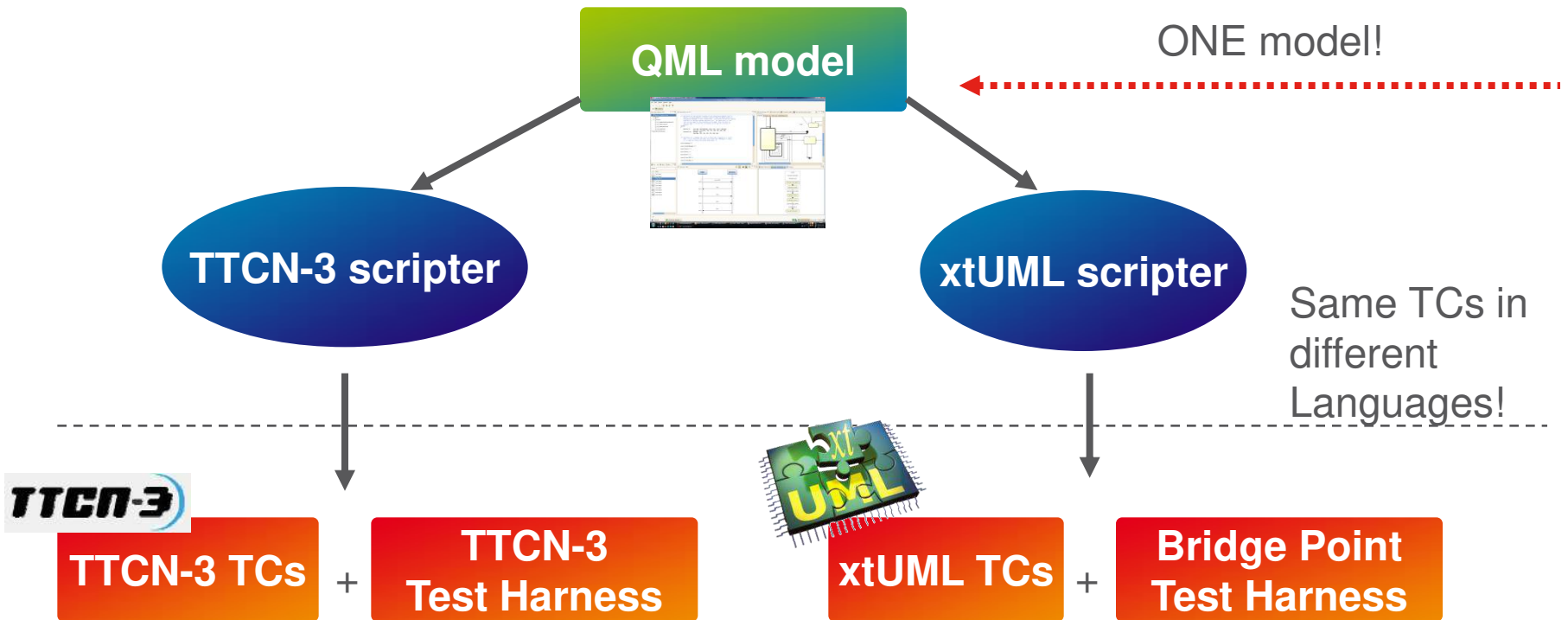
Media Player



SIP Terminating



Function Testing with MBT



› Test Model

- Test goal specific
- High level

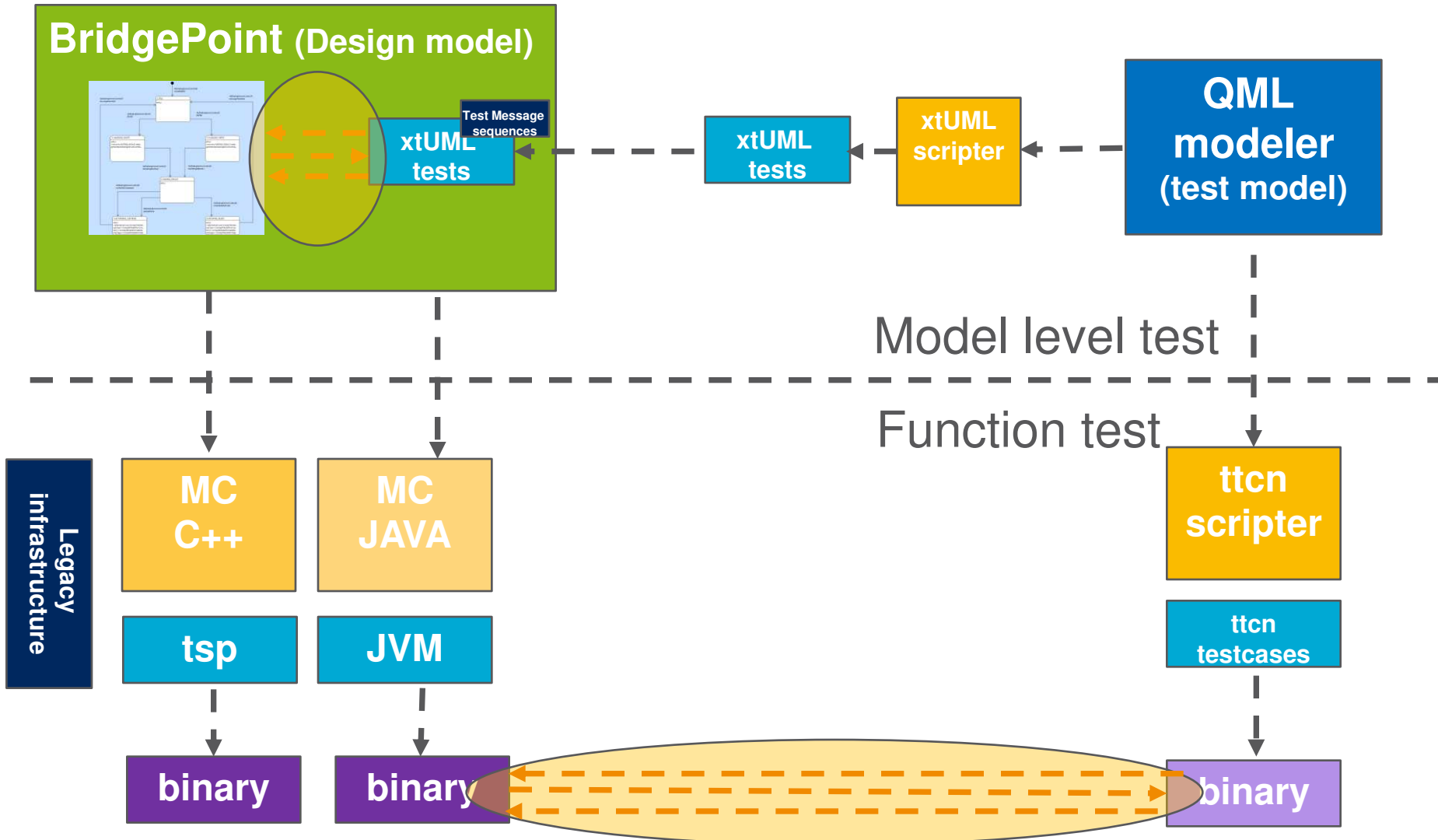
› Test Harnesses

- Provides adaptation towards the SUT

› Scripters

- Used to generate the code that realize the tests
- TTCN-3 scripiter, OAL (xtUML) scripiter

Tool chain Synergy



Challenges and solutions



- › Integration
 - Deciding the model boundary
 - Adaptation to legacy frameworks' APIs
- › Version control and collaborative development
 - Model merging problems
 - › Graphical conflicts
 - › Not resolvable conflicts: e.g. structural changes
 - Solutions
 - › Tools for automated merge
 - › Work separation

Challenges and solutions



- › Resistance at the receiver organization
 - Remain enthusiastic
 - Working examples are the best evidences
 - Be patient, let them do it themselves
 - If one turns, the rest will follow
 - Expect slow start, they have to learn
- › Consultancy
 - Orthogonal knowledge areas
 - Tailoring for the different needs
 - Providing guidelines (e.g. Do not touch the generated code)
- › Tools are expensive
 - Work separation

benefits



› Modeling

- Raising the abstraction level
- Thinking before coding
 - › Use-cases -> Test strategy
 - › Fixing it on the drawing board or fixing it on-site
- Early testing
- Enables automation

› Test driven development

- Builds confidence
- Facilitates discussion



ERICSSON