Agenda

• Platform Maturity & Skills
  - Survey Results
  - Recommendations
  - Recommended Platform Maturity Levels

• Technology
  - Survey Results
  - Recommendations
Survey response was decent but incomplete

• 15 Project Teams responded:

MSB, VFC, Holmes, ONAP CLI, APPC, VID, Policy Framework, Portal Platform, Documentation, DCAEGEN2, VNFSDK, CCSDK, SDNC, SDC, VF-C

Thank you to the responding projects!
Platform Maturity & Skills
Except for resiliency & scalability, most projects are at early stages in platform maturity requirements (aka S3P).

**NOTE:** A low level assessment for Amsterdam is not unexpected, since many categories require establishing a baseline of capability and improving in future releases.
Projects were generally confident in their teams’ skills…
... but comments revealed some needs

- Overall
  - Need more resources, particularly with Kubernetes and integration skills

- Performance & Stability
  - Need resources and assistance in doing performance and stability testing

- Resiliency & Scalability
  - Need more resources & techniques for failover and scaling

- Security
  - Need assistance in how to achieve CII Badging (most requested)
Recommended Platform Maturity Levels for Beijing* (1/2)

<table>
<thead>
<tr>
<th>Area</th>
<th>Priority</th>
<th>Min. Level</th>
<th>Stretch Goal</th>
<th>Level Descriptions (abbreviated)</th>
</tr>
</thead>
</table>
| Performance| Low/Med  | Level 1 – closed-loop projects Level 0 – remaining projects Level 1 – remaining projects | Level 1 – remaining projects | •0 -- none  
•1 -- baseline performance criteria identified and measured  
•2 & 3 – performance improvement plans created & implemented |
| Stability  | Medium   | Level 1              | Level 2 – run-time projects | Level 2 – run-time projects  
•0 -- none  
•1 – 72 hour component level soak w/random transactions  
•2 – 72 hour platform level soak w/random transactions  
•3 – 6 month track record of reduced defect rate |
| Resiliency | High     | Level 2 – run-time projects Level 1 – remaining projects Level 2 – remaining projects | Level 3 – run-time projects Level 2 – remaining projects | Level 3 – run-time projects  
•0 -- none  
•1 – manual failure and recovery (< 30 minutes)  
•2 – automated detection and recovery (single site)  
•3 – automated detection and recovery (geo redundancy) |

*Adapted from AT&T Ops Team presentation (Lee Breslau):
[https://wiki.onap.org/display/DW/Contributions?preview=/8225716/20087412/ATT%20Review%20of%20ONAP%20Carrier%20Grade%20Requirements.pptx](https://wiki.onap.org/display/DW/Contributions?preview=/8225716/20087412/ATT%20Review%20of%20ONAP%20Carrier%20Grade%20Requirements.pptx)

# Recommended Platform Maturity Levels for Beijing* (2/2)

<table>
<thead>
<tr>
<th>Area</th>
<th>Priority</th>
<th>Min. Level</th>
<th>Stretch Goal</th>
<th>Level Descriptions (abbreviated)</th>
</tr>
</thead>
</table>
| Security     | High     | **Level 1** - 70% of projects; non-passing meet 80% of requirements  
Cryptographic – all projects | **Level 2** | • 0 – none  
• 1 – CII Passing badge  
• 2 – CII Silver badge; internal communication encrypted; role-based access control and authorization for all calls  
• 3 – CII Gold |
| Scalability  | Low      | **Level 1** – run-time projects  
**Level 0** – remaining projects | **Level 1** | • 0 – no ability to scale  
• 1 – single site horizontal scaling  
• 2 – geographic scaling  
• 3 – scaling across multiple ONAP instances |
| Manageability| High     | **Level 1**                                      | **Level 2**  | • 1 – single logging system across components; instantiation in < 1 hour  
• 2 – ability to upgrade a single component; tracing across components; **externalized configuration management** |
| Usability    | Moderate | **Level 1**                                      | **Level 2**  | 1 – user guide; deployment documentation; API documentation  
2 – UI consistency; usability testing; tutorial documentation |

*Adapted from AT&T Ops Team presentation (Lee Breslau):  
https://wiki.onap.org/display/DW/Contributions?preview=/8225716/20087412/ATT%20Review%20of%20ONAP%20Carrier%20Grade%20Requirements.pptx  
DRAFT
Recommendations -- Platform Maturity Requirements & Skills

• **Security** – Security subcommittee enact enablement plan to help teams reach appropriate badge level; identify any assisting technology
  - *Potential new projects*: Vault CA project

• **Resiliency/Scalability** – Multiple projects to lead technology rollout across projects for higher resiliency; supervised by architecture subcommittee
  - *Potential new projects*: CHAP, OOM enhancements, MSB enhancements

• **Performance & Stability** – Integration team to create process for teams to do performance and stability testing. Identify needs for labs and tools
  - *Potential new projects*: Benchmarking

• **Manageability** – Logging Enhancements Project team to create enablement & tooling for consolidated logging & transaction tracing service
  - *Potential new projects*: Distributed K-V store

• **Skills** -- Reach out to board for help on obtaining more resources
Technology
Java is the predominant programming language with many others in use.

NOTE: One project listed 8 different programming languages in use!
Spring is the primary coding framework
A wide variety of data storage mechanisms are in use.

The diagram shows a comparison of different data storage methods, including SQL and NoSQL systems. The methods are categorized into SQL (MySQL, PostreSQL, MariaDB) and NoSQL (Cassandra, ODL MDSAL, JCSCache, Redis). The vertical axis represents the usage or popularity of each method, with values ranging from 0 to 3.5.
DMaaP is the primary messaging system

== could be DMaaP, since DMaaP is based on Kafka
A wide variety of parsers are in use for TOSCA and XML/JSON
Logging Frameworks are pretty dispersed

Logging Frameworks

- Logback
- log4j
- SLF4J
- EELF
Most projects use Angular for their UI Framework
Recommendations -- Technology

• Continue adoption of existing shared services:
  - Messaging (DMaaP)
  - Service Discovery/Routing (MSB)
  - Logging Enhancements

• Consider shared services for some categories:
  - Data storage
  - TOSCA parser

• Architecture subcommittee to identify *preferred* technologies in the key remaining areas
  - Preferably, teams treat migration as technical debt to be worked
Next Steps

• **Formalize Platform Maturity Level recommendations**
  - Gather feedback over next 2 days
  - Vote in TSC meeting on Wednesday

• **Determine best governance for Software Architecture**
  - *Software Architecture Coordinator* -- works with PTLs and Use Case, Security & Architecture Subcommittees
  - *Software Architecture Subcommittee* -- consisting of software architects
  - *No Software Architecture coordinator or subcommittee* – can be handled by projects and existing subcommittees
Questions?