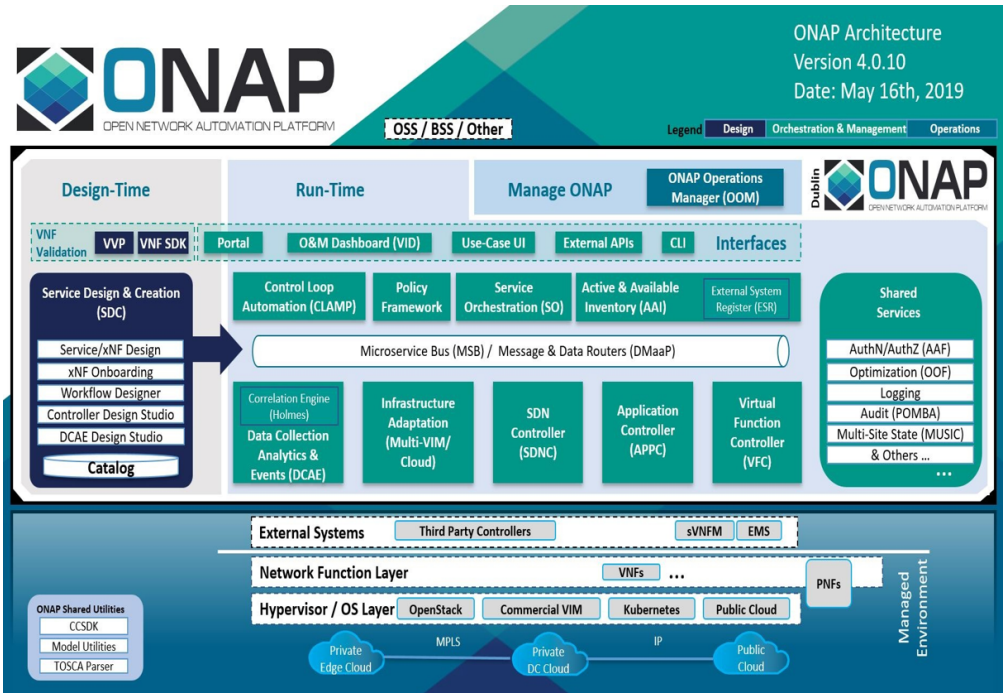
ONAP Architecture Team Process (Copy)

The following diagram illustrates the overall Architecture Process. It shows the basic stages of a release, going from M0 (release kickoff), to M2/M3 (API Freeze) and to code delivery in M4.

It illustrates 5 stages that the Architecture teams are concerned with, the Functional Architecture requirements, to component architecture reviews and architecture improvements.

FUNCTIONAL ARCHITECTURE - The functional architecture is the high-level architecture overview diagram for all of ONAP. This was the Functional Architecture diagram on May 16, 2019:



COMPONENT ARCHITECTURE - The component architecture are the platform components. Examples of platform components are SO, A&AI, CCSDK, SDN-C. Each release there may be architecture impacts from the platform components

REQUIREMENTS ARCHITECTURE - These are architecture impacts coming from the requirements and use case work in a release that may impact the functional architecture, platform architecture, or may need architectural guidance.

ARCHITECTURE ENHANCEMENTS - Architecture enhancements are secondary architectural enhancements that are worked during a release. These may include documentation enhancements, landing page enhancements, architecture component description work, flow descriptions and process work (this wiki page!)

Some key things the happen at the milestones:

M0 - Functional Architecture Proposals, Component Architecture Proposals, Requirements Architecture Proposals

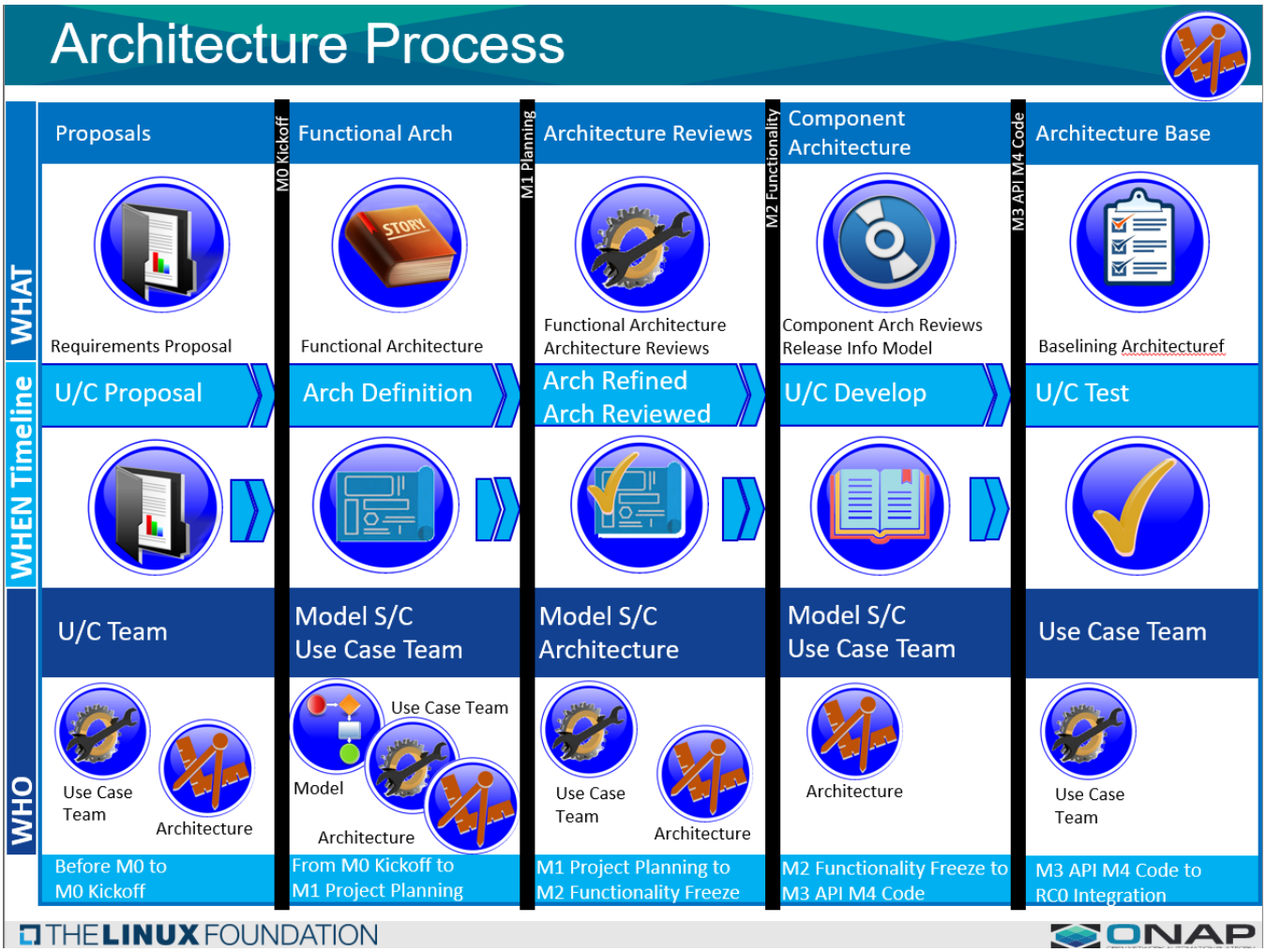
M1 - Functional Architecture Review, Component Architecture Requirements, Requirements Architecture Requirements, Architecture Enhancements Proposals

M2 - Component Architecture Reviews & Base-line, Requirements Architecture Review & base-line, Architecture Enhancements Reviews

M3 - Functional Architecture base-lined, Information Model sync, Architecture Enhancements review & base-lined

M4 - Test Case review and Architecture consistency.

Drop	Functional Architecture	Component Architecture	Requirements Architecture	Architecture Enhancements	Info Model	Test Cases
M0	FA - Proposal Architecture Team	CA - Proposal from PTLs	FcA - Proposal Dev U/C teams	AE - Proposals Architecture Team		
M1	FA - Review	CA - Requirements	FcA - Requirements	AE - Requirements		
M2		CA - Review	FcA - Review	AE - Review		
M3	FA - Baseline	CA - Baseline	FcA - Baseline	AE - Baseline	Sync	
M4						Sync



Release Planning (legacy)

Frankfurt Release M3 API Freeze Milestone Checklist

Use Case Teams Process

- **M0**
  - **ARCHITECTURE SUB-COMMITTEE** - The following are M0 activities with the Architecture Sub-committee. Release content defined.
    - **#1 FUNCTIONAL ARCHITECTURE (Proposals)** - The functional reference architecture is the high-level architecture overview diagram for all of ONAP. Enhancements to the functional architecture may be driven by new project proposals, updates to the diagram, and architectural changes that may be planned for the release. At M0 impacts to the functional architecture are proposed.
    - **#2 COMPONENT ARCHITECTURE (Proposals)** - The component architecture impacts originate from the ONAP platform components. Examples of platform components are SO, A&AI, CCSDK, SDN-C. Each release there may be architecture impacts from the platform components. At M0, component architecture impact proposals are submitted. Architecture reviews are setup & scheduled for component architecture impacts/proposals. Component architecture diagrams would be updated.
    - **#3 REQUIREMENTS ARCHITECTURE (Proposals)** - These are architecture impacts coming from the requirements and use case work in a release that may impact the functional architecture, platform architecture, or may need architectural guidance. At M0, the requirements and use cases are being proposed for the release. And an early assessment of which ones that might impact the architecture should be considered, and they made translate into requirements architecture proposals. Architecture reviews are setup & scheduled for requirements architecture impacts/proposals.
    - **#4 ARCHITECTURE ENHANCEMENTS (Proposals)** - Architecture enhancements are secondary architectural enhancements that are worked during a release. These may include documentation enhancements, landing page enhancements, architecture component description work, flow descriptions and process work. At M0, initial proposals are submitted to the architecture sub-committee.
  - **WIKI LINKS References for the Use Case Teams at M0**

WIKI LINKS REFERENCE AT M0

M0	Description	Wiki Link
1	**Functional Reference Architecture**	Architecture
2	**Architecture Portal Page**	https://safratech.net/onapdocs/
3	**Release Architecture Page**	All of the architecture reviews that were conducted:
4	**Requirements Proposals**	Requirements subcommittee (is merged into ARCCOM) Guilin release - functional requirements proposed list
Mod	**Modeling Release Planning Page**	ONAP R7 Modeling High Level Requirements

- **Modeling team** - At MO the Modeling S/C does **MODEL PLANNING**. The planning develops into “High Level Info-model Requirements”. These High level info-model requirements fall into 3 categories:
  - #1: **NEW USE CASES** - items from the expected Use Cases in the release (Scope of modeling, continuing, introducing, standards updates). The Use Case Teams should engage the modeling team to propose new requirements into their release planning page.
  - #2: **REFINING EXISTING MODEL** - Refining Existing info-model that has not made it to the information model. Previously designs that need to be added into information model.
  - #3: **FORWARD LOOKING WORK (FLW)** - Modeling future forward looking requirements.
- **USE CASE TEAMS** At M0, the Use Case teams are working on the following things:
  - #1: **BUSINESS DRIVER TEMPLATE** - The Use Case Teams are specifying their business drivers via the template: Business Driver Template for Use Cases
  - #2: **REQUIREMENTS SUB-COMMITTEE** - Develop their proposals for the release to the Requirements Sub-committee: Requirements subcommittee (is merged into ARCCOM)
  - #3: **REQUIREMENTS RELEASE TRACKING** - Requirements put release requirements page. Example wiki: Guilin release - functional requirements proposed list
  - #4: **USE CASE DEFINITIONS** - Use Case team fill out the Template detailing their Use Case: Use Case Tracking Template
  - #5: **INTENTION TO PARTICIPATE** - Teams can indicate their corporate intention to participate
  - #6: **RELEASE TRACKING** - Each release has a release tracking page. The page can be found here: Release Planning
  - #7: **PROJECT SUBMISSION, PROPOSAL, PLANNING** - The TSC coordinates the project submission, proposal and planning. The TSC reviews and gives disposition on submitted proposals. These are tracked at the Release Planning page.
- **PTL** - High level release scope from PTLs (understand from PTL which ONAP components are expected to have updates)
- **PTL** - The Use Case teams should attend the PTL planning meetings if there are expected to be requirements impacts for your use case. It is also where the Release Planning page is developed by the PTL team.

M1

- **ARCHITECTURE SUBCOMMITTEE leading up to M1** -
  - **#1 FUNCTIONAL ARCHITECTURE (Requirements)** - The functional reference architecture is the high-level architecture overview diagram for all of ONAP. Enhancements to the functional architecture may be driven by new project proposals, updates to the diagram, and architectural changes that may be planned for the release. At M0 impacts to the functional architecture are proposed.
  - **#2 COMPONENT ARCHITECTURE (Requirements)** - The component architecture impacts originate from the ONAP platform components. Examples of platform components are SO, A&AI, CCSDK, SDN-C. Each release there may be architecture impacts from the platform components. At M0, component architecture impact proposals are submitted.
  - **#3 REQUIREMENTS ARCHITECTURE (Requirements)** - These are architecture impacts coming from the requirements and use case work in a release that may impact the functional architecture, platform architecture, or may need architectural guidance. At M0, the requirements and use cases are being proposed for the release. And an early assessment of which ones that might impact the architecture should be considered, and they made translate into requirements architecture proposals.
  - **#4 ARCHITECTURE ENHANCEMENTS (Requirements)** - Architecture enhancements are secondary architectural enhancements that are worked during a release. These may include documentation enhancements, landing page enhancements, architecture component description work, flow descriptions and process work. At M0, initial proposals are submitted to the architecture sub-committee.
- **WIKI LINKS References for Architecture at M1**
  - The following table has the links for important M1 architecture pages

WIKI LINKS REFERENCE AT M1

M1	DESCRIPTION	WIKI LINK
1	**Architecture Portal Page**	https://safratech.net/onapdocs/
2	**Architecture Release Page**	
3	**Milestone Checklist**	Deliverables for Planning Milestone Checklist Template
4	x	

- **USE CASE TEAMS** - The Use Case Teams prepare their proposals. The Use case teams are defining their requirements and detailed proposals.

        - #1 **RELEASE PLANNING TEMPLATE** - Use Case teams use the release planning template to help you think about your scope, minimum viable product, architecture, risks, and epics. The release planning template can be found here: Release Planning Template
        - #2 **MILESTONE CHECKLIST** - The milestone checklist template are used by the Use Case teams. This checklist can be found at this wiki link: Deliverables for Planning Milestone Checklist Template
        - #3 **PROJECT DELIVERABLES** - are defined including functional architecture, scope, dependencies.
        - #4 **MODELING TEAM WORK** - The model subcommittee team becomes aware of the use cases for the current release.
        - #5 **EARLY DATA MODEL CONCEPTS** - Because the information model feeds the data models, the Use Case teams should take into account the new updates in the information model as a basis for their data model.
    - **Project & Release planning tables**
        - The architecture checklist is from the Deliverables for Planning Milestone Checklist Template

AREA	CHECKLIST ITEM	LINK
Architecture	Ax	
	Axr	
	Ha	
x	Is tx	
	Ha	
	Is txp	

    - **Modeling team** The info-model plan is established by the modeling team which summarizes the modeling requirements for a release. The model planning follows a template that is worked by the team. Info-model updates begin. An example template for R6 (Frankfurt) can be seen at this Wiki: ONAP R6 Modeling High Level Requirements.
        - #1: **SYNC** - The Use Case teams need to engage the Modeling Sub-committee to make the team aware of model impacting use cases. The modeling team should also become aware of at a high-level what impacts a use case might have. The use cases also need to get into the ONAP Modeling High-level requirements planning page.
    - **Components (PTL)** - Each of the ONAP platform components (e.g. A&AI, SO, Controllers, SDC etc) may be impacted by new use cases. Having the use case leads engage PTLs.
        - #1: **COMMITMENT & TRACKING** - The data model developed by the use case teams eventually serve as the basis for API changes. Platform components need to update APIs based on new requirements, use cases and features. Requests to components need to be tracked & commitment by the PTLs and components. Ideally the PTLs and component leads should be engaged by the Use Case teams. SDC & A&AI often have more high-running modeling impacts than some of the other components. The modeling team members could attend some of the component calls to raise awareness. Identifying and tracking a modeling impacting item so they aren't lost. An issue impact matrix and tracking page could be developed to track issues (and maybe a Jira ticket).
        - #2 **RELEASE TRACKING PAGE** - The release tracking page also tracks the platform components.

APIs producing consuming at architecture review

Modeling S/C to understand - Identify APIs information that retreiving updating across those APIs.

Esp. anything NEW. Info-Model impacts exposing new stuff on an API indicating new information being shared should have RELATED Modeling

Here's new INFO on expensing

Devloping updates on an API. related info-modeling activity.

M2

- **ARCHITECTURE SUBCOMMITTEE leading up to M2** -
    - **#1 FUNCTIONAL ARCHITECTURE (Architecture Reviews)** - The functional reference architecture is the high-level architecture overview diagram for all of ONAP. Enhancements to the functional architecture may be driven by new project proposals, updates to the diagram, and architectural changes that may be planned for the release. At M0 impacts to the functional architecture are proposed.
    - **#2 COMPONENT ARCHITECTURE (Architecture Reviews)** - The component architecture impacts originate from the ONAP platform components. Examples of platform components are SO, A&AI, CCSDK, SDN-C. Each release there may be architecture impacts from the platform components. At M0, component architecture impact proposals are submitted.
    - **#3 REQUIREMENTS ARCHITECTURE (Architecture Reviews)** - These are architecture impacts coming from the requirements and use case work in a release that may impact the functional architecture, platform architecture, or may need architectural guidance. At M0, the requirements and use cases are being proposed for the release. And an early assessment of which ones that might impact the architecture should be considered, and they made translate into requirements architecture proposals.
    - **#4 ARCHITECTURE ENHANCEMENTS (Architecture Reviews)** - Architecture enhancements are secondary architectural enhancements that are worked during a release. These may include documentation enhancements, landing page enhancements, architecture component description work, flow descriptions and process work. At M0, initial proposals are submitted to the architecture sub-committee.
- **Use Case Team Engagement** -
    - **FUNCTIONAL TEST CASES** - Each Project team has defined and documented their Functional Test Cases.
    - **API DOCUMENTATION** - The Use Case Teams have identified the API that they will be introducing in the current release and at least started to document that API changes that are associated with their use case. The APIs need to be documented and available in the Use Case Wiki.

        - **VNF REQUIREMENTS** - Base set of impacts to VNF Requirements identified by use case/ project.
        - **INFORMATION & DATA MODEL DEVELOPMENT** - Discussion Info Model & Data model development by Use Case team with input from the Modeling sub-committee. Active discussion and interaction between Use Case Team and the Modeling S/C to make sure that the information model and the data model development are in lock-step. The modeling sub-committee will communicate the clean release information model as a refining input to the development of the data model for use by the Use Case Teams on specific projects.
        - **INFORMATION & DATA MODEL REVIEW** - Reviews of Data Model with Project (Use Case) Teams. The Data Model is being reviewed by the Use Case Teams with inputs from the Modeling S/C by bringing the developing data model (in the discussion state) to the modeling S/C. It would not be feasible to for the members of the modeling S/C to attend all of the various U/C meetings; although one-off sync-ups might occur in this stage. For those U/C that have significant data modeling work, it would be advised that that U/C team reserves a slot in the modeling S/C meeting(s) to present their data modeling changes and information flows so that the modeling S/C team can advise the U/C team as they develop their data model.
        - **MAPPING BETWEEN INFORMATION & DATA MODEL** - Mapping of information model and the data model is also done between the modeling S/C and the Use case teams. This might happen in the project teams, or on the modeling S/C calls.
        - **CROSS REFERENCING JIRA TICKETS** - The modeling S/C uses Jira tickets to track activities; and the Use Case teams also use Jiras to track platform work, modeling work, epics & stories. So it would be smart to link or associate relevant Jira tickets together.
        - **JOINT REVIEWS** - The Data model should be reviewed with the Modeling S/C. Data model being developed by the component team is using the component model as input.
        - **FUNCTIONALITY IDENTIFIED** - The release intended functionality has been identified, agreed to and frozen. No new visible functionality is to be added to the current ONAP release at the point of M2.
    - **MODELING SUBCOMMITTEE** -
    - For the RELEASE Information Model these are the activities that the Modeling sub-committee is engaged in leading up to M2.
        - **RELEASE INFORMATION MODEL** (Starting Point) - The release starts with a clean *release information model* from the PREVIOUS release (with all of its attendant contributions). Then new contributions of the current release are considered (*see below the process for handling each specific contribution*). Potentially a snapshot of the papyrus model and posted into the current release. The RST documentation that only contains things in the current release or everything that is approved.
        - **INFORMATION MODEL FREEZE** - The aggregate / release information model for the release is approved by association with the fragment/ component reviews. Each of the fragment (contributions) are individually approved, thus there is not a "re-approval" or approval of the entire aggregate (release) information model. Editorial clean-up such as misalignments, typos, or sections that were not put in proposal, fixing the template for GenDoc.
        - **RELEASE INFO MODEL DECLARED CLEAN** - After component reviews have concluded and release info model freeze by the modeling S/C the info model is called the "clean model" in this phase. At this point, the Use Case teams that are developing the Data Model can be pretty certain that the information model will be usable as shown. The diagrams and model wiki pages will indicate that this is a clean model. Put into the information model for that release. Unfinished contributions are postponed or discussed further.
    - **DISCUSSION OF CONTRIBUTIONS** - Each contribution discussed according to following process. This is where refining of each of the contribution models occurs by the Modeling Sub-committee (S/C). The release information model is not separately tracked, composed, updated, or released in this period of time. But, rather, each individual contribution has its own Wiki. Thus, for each contribution:
        - **CONSIDER CONTRIBUTION** - START: Input Contribution (verb Consider) END: Contribution in Discussion State
            - An individual model contribution is a model that will eventually be a part of the total release information model. It is generally a self-contained model which depicts a particular capability or function of the system. This contribution starts as a "*input contribution*" and undergoes *consideration* by the modeling sub-committee. *Consideration* means that the modeling S/C is entertains & assesses if the *input contribution* should be accepted into the current (or a future release) by weighing the contribution against its relevance and the available resources (modelers) in the release. If the team thinks that the contribution is not ready for the current release that contribution will be put into a lower-priority and worked if there are no other contributions to be considered as they would take higher priority. Thus, the contribution would not necessarily be rejected, but would get attention as time allows.
        - **REVIEW & REFINE CONTRIBUTION** - START: Contribution in Discussion State (verb Reviewing & Refine) END: Contribution in Discussion state
            - The contribution undergoes *reviewing & refining* during the *discussion state*. *Reviewing & refining* means that the modeling S/C is discussing the modeling, and updating the contribution based on feedback and comments from the modeling team. Each contribution can be reviewed and refined independently and concurrently with other contributions. Things in the discussion state are classes, attributes and relationships are tagged as IISOMI *experimenta*
              .
        - **FINAL CALL FOR COMMENTS & INITIATE POLLING** - START: Contribution in Discussion State (verb Approving/Poll) END: Contribution in Discussion state. The modeling team issues a call for comments and poll for each contribution.
        - **APPROVING CONTRIBUTION** - START: Contribution in Discussion State Post-Poll (verb Approving) Contribution in Clean State
            - After the poll has concluded, the contribution has finished the *approval* process. The contribution is now considered to be in the *clean state*. The items that are in the IISOMI *experimental* state get promoted to a *preliminary* state. A gendoc is generated and put on the wiki page. The gendoc would be translated and published on the readthedocs site.

- **Components (PTL) Engagement** - ONAP Platform Teams (A&AI, SO, SDC etc) review clean Information Model impacts for the release.
    - **FEEDBACK** - Component platform work can feedback to the Modeling S/C for updates to the information model during the refining the info model phase and should also provide input during the review. Modeling S/C should take into account component platform updates vis-a-vis the Use Case and modeling requirements for the release.
    - **SOCIALIZATION** - The socialization of the clean release information model should include updates for the PTLs. The platform PTLs must become aware that the clean release information model has gone to approval. The PTLs also attend the TSC. An email to the PTLs. Possibly a joint call with the PTLs in attendance might help to socialize the information model. Because this is a major milestone of the modeling S/C. Perhaps a modeling notification email distribution list could be made that would send major updates from the modeling S/C and that would not flood notifications from the modeling team. An email announcement of polls, in this case the baseline of a clean release information model.

M2 CHECKPOINTS & COLLABORATION

- **M3**
  - **Use Case Engagement** -
    - **API Freeze** - M3 is characterized by the API freeze. The main thing that happens at M3, the API is frozen by the Use Case Teams.
    - **Data Model Freeze** - Developers identify a problem in the data model which affects the information model.
      - **SYNC UP** - since the Modeling S/C is familiar with the info-model; the Use Case Teams should present at the Modeling S/C their proposed data model that might be frozen so that the modeling S/C can assess it to see if might have impact to the Info-model. There should be some collaboration or check-point at M3 to discuss and potential ripple affects back to the information model.
      - **DATA MODEL IMPACTING INFO MODEL** - If changes in the Data Model impact the information model, those changes need to be worked by the model S/C. The Modeling S/C would evaluate the change to the Information model and possibly make updates.
      - **USE CASE TEAMS INDICATE CHANGE** - The Use Case teams may have enough knowledge of the info-model that they identify a data model change that may impact the information model. This presumes that the use case teams know that their changes in a data model may have impact to the information model.**B**
      - **CONSIDER DATA MODEL** - START: Input Data Model > verb Consider > END: Data Model in Discussion State
        - The data model is a model that is used in a Use Case and is based on the Information Model. It is generally a self-contained model which depicts a particular capability or function of the system. The data model starts as a "*input data model*" and undergoes *consideration* by the Use Case teams*. Consideration* means that the Use Case teams is entertains & assesses if the *input data model.* If the Use Case teams think that the contribution is not ready for the current release that contribution it might postponed. It would be noted in the Release Management Project page as such.
      - **REVIEW & REFINE CONTRIBUTION** - START: Data Model in Discussion State > verb Reviewing & Refine > END: Data Model in Discussion state
        - The data model undergoes *reviewing & refining* during the *discussion state*. *Reviewing & refining* means that the Use Case Teams are discussing the data model and updating their data model based on feedback and comments from the Use Case team and modeling team. Each data model can be reviewed and refined independently and concurrently with other use case projects. Things in the discussion state are classes, attributes and relationships are tagged as IISOMI *experimental*.
      - **MODELING S/C ENGAGEMENT** - The Use Case teams may wish to solicit the opinion of the modeling S/C and present their data model for discussion and socialization.
      - **FINAL CALL FOR COMMENTS & INITIATE POLLING** - START: Data Model in Discussion State > verb Approving /Poll > END: Data Model in Discussion state
        - (a) **FINAL PRESENTATION** - When the data model has gotten to a point where the use case team feels that it can start to undergo the approval process, the data model is brought one final time the use case team.
        - (b) **FINAL CALL FOR COMMENTS** - After that, a final call for comments is issued by a *use case lead* to the modeling team whereby final thoughts & input can be given. This final call for comments signals that the discussion is wrapping up for this contribution and will soon go to a poll.
        - (c) **INITIATING POLL** - After final call and no further outstanding comments exist, the contribution is brought to a poll by a *use case lead*. A poll is created whereby use case team members can give the contribution a vote of "*yes*" or "*no*".
      - **APPROVING CONTRIBUTION** - START: Data model is in Discussion State Post-Poll > verb Approving > Data model in Clean State
        - After the poll has concluded, the data model has finished the *approval* process. The data model is now considered to be in the *clean state*. The items that are in the IISOMI *experimental* state get promoted to a *preli minary* state. A gendoc is generated and put on the wiki page. The gendoc would be translated and published on the readthedocs site. #@# do data models also use ISOMII states?
    - **Component Data Model Final** - x.
    - **RECONCILE** - Reconciling the info-model with the data-model. M3 checklist. (API Freeze)
    - **M3 CHECKLIST** - The M3 check list is used. This is a vehicle to engage the Use Case (project teams) and reconcile the Use Case Teams with the modeling S/C team's work. The Check list can be found here: Proposed M3 Checklist modeling updates discussion
  - **M3 MODELING SUBCOMMITTEE ACTIVITIES** -
    - **REFINEMENTS TO THE RELEASE INFO MODEL** - The Release Information Model is clean at M3. It is considered "base-lined" and "final", hence it is marked clean. Though, updates can still happen to the release information model and the model contributions therein. This means that certain *elements* within the model(s) could go to back to an experimental state. Note that only certain elements (e.g. attributes, ranges) are likely to go to the experimental state NOT the entire contribution. More often though, new additions could be added to a contribution model. In general, there would likely be just minor tweaks on the model. So when a contribution is clean it has to be at least preliminary. A contribution cannot be clean and experimental. Clean has a relationship to the IISOMI states. For an entity to be clean it must be either preliminary or mature (see the IISOMI state diagram link).
      - **IISOMI STATES** - A link to the IISOMI state diagram can be found here: Stereotypes
      - **NEW ADDITIONS** - A contribution model could be clean, but things added afterwards and those elements would come in as experimental.
    - **STILL IN PROGRESS ITEMS IN RELEASE INFO MODEL** - It is possible that as the modeling team enters the M3 milestone that there are still some things in progress, that are expected to be in the current release. They might still be marked *experiment al* even though the release information model is *clean*. Thus, to open item are continue to be work; and it is expected it would not affect software if code were already associated to it.
      - **ITEMS IN DISCUSSION** - e.g. when root contribution was done, with root party is an example as it was not agreed to, we made the decision to leave that experimental until a future date. There were aspects agreed, and other things left experimental to pursue in the future. The main contribution was split. These parts everyone agreed with and these part left experimental which would be taken up in a future contribution and re-discussed. This would likely occur at M2, and they might be discussed at M3. There was a conscious decision and agreement by the modeling team that the parts of the model still open would be pushed to the next release. So only theoretical discussion would happen at this point of how to proceed in the future release.

            - **FUTURE WORK** - Things originally planned to be in a release could potentially transform into future work items. Some modeling work could be pushed to the next release if need be, if it is decided that it could not be completed in the current release.
        - **FUTURE WORK** - Future work is typically identified as such at the start of a release at M0 in the release modeling planning page. Future Work can still proceed. For example, in R6 the geo-location modeling work is not tied to any active development yet. The location work is a good example of work that was worked in ADVANCE of when it is expected to be used (Future Work). It is also possible that some of the future work is building upon a foundation of work that had already been started (or was looked at) or implemented in a prior release.
            - **DEFER WORK** - It might be decided the the future work could be deferred to the next release. On the current modeling high level requirements page to indicate that a particular future work has been deferred to a future release. In order not to lose the activity, it would be expected that it would be rolled into the *next release's Modeling High-Level Requirements*.
            - **CONTINUE WORK** - Future Model work may continue to proceed in the current release..
        - **DOCUMENTATION AFTER IMPLEMENTATION IN PRIOR RELEASE WORK** - This type of work is the model is catching up to already implemented software. It has already been identified as something that would be worked on for that release at M0. It is expected that it wouldn't immediately impact the current software. However, it may be extended eventually to incorporate new work. The way to proceed with this category of work is handled the same way as the future work (i.e. Defer or to Continue the work) given modeling improvement recommendations on how better to model a given concept.
        - **M3 CHECKLIST** - The *M3 check list modeling updates discussion* is used by the modeling sub-committee. It is used as a vehicle to engage the Use Case (project teams) and reconcile the Use Case Teams with the modeling S/C team's work. See also the Use Case Team Engagement (section below). The Check list can be found here: Proposed M3 Checklist modeling updates discussion
    - **Architecture Engagement** -
        - **S-P** - B-
    - **Components (PTL) Engagement** -

- M4
    - **Code Freeze**
    - **Kickoff Information Model Requirements for Next Release**
    - **READ THE DOCS** - (M3 or M4?) The model editor provides a final gendoc word document which serves as the basis for what will be incorporated into the readthedocs. The read the docs can be found here: https://onap.readthedocs.io/en/latest/index.html . The word document is fed into some tools which generates the readthedocs output. The gerrit master model is periodically updated, and a snapshot of the eclipse/papryus model is taken and that is called the release model.
    - **DOCUMENT GENERATION** - The RST documentation that only contains things in the current release or everything that is approved.
    - **PAPYRUS GENERATION** - The Papyrus snapshot is generated. The RST document is created. The readthedocs documentation is generated.
    - Note that the papyrus model includes what was/had accepted into the previous release and also anything that is still a work in progress.
    - **Architecture Engagement** -
        - **S-P** - B-
    - **Use Case Engagement** -
        - **D-T** - D-p.
    - **Components (PTL) Engagement** -
- RCx
    - **Runtime Compliance**
- Observations
    - **Establishes and Evolves a Common Model**
    - **Project (Component) Team Involvement in Modeling Solution**
    - **Governance of Common Model and Corresponding Component Models**
    - **Update possible in M3 and M4 (bug fixes) per exception process**

Artifacts

- Information Model Artifact Contains
    - **Classes**
    - **Relationships with Multiplicity**
    - **Important Attributes with Multiplicity**
    - **Definitions**
    - **Data Types**
    - **Feed to Data Dictionary**
    - **Tooling - Papyrus with GitHub**
- Component Data Model Artifacts (Implementation Specific)
    - **Component Data Model**
        - **Contains objects, attributes, & relationships (more detail than information model)**
    - **Mapping to Information Model**
    - **Feed to Data Dictionary?**

- API Artifacts
    - **API Model**
    - **Mapping to Information Model**

New Roles – Model Governance

- **Information Model**
    - **Internal Committers**
    - **Internal Approvers**
    - **Impacted Project (Component) Approvers**
    - **Impacted API Approvers**
    - **Architecture Group Approvers**
- **Component Data Models**
    - **Internal Committers**
    - **Modeling Team Approvers**
    - **Architecture Approvers**
    - **Impacted API Approvers**
- **API Definitions**
    - **Modeling Team Approvers**
    - **Impacted Project (Component) Approvers**
    - **Architecture Approvers**

Benefits

- **Establishment and Evolution of a Common Model (Model Consistency)**
- **Continue Move Toward a Model Driven Design**
- **Improve Data Quality**

Modeling S/C, Use Case Team and Architecture team touch points, interactions and cooperation:
blocked URL

SUPPORTING DOCUMENTS

DOC	FILE
Way of Working (Modeling S/C, Use Case Teams, Architecture)	? Unknown Attachment