Interview Guide

Background information

General Experience and Background

1. How many years of professional experience do you have in software engineering/developing software?
2. For which domain do you develop software?
3. What kind of software systems do you develop?

Experience with formalization methods

1. Do you have experiences with formal methods?
    a. if yes, which ones? for example Behavior Driven Design, Architecture Description Languages, Object Constraint Language
    b. if you have experience, how much experience (in years, to which extent) do you have with formal methods?
2. What is your attitude regarding formal methods?
    a. are they necessary, helpful, useless?
3. Do you have experiences with architecture rule formalization and validation tools (e.g. architecture conformance checking tools)?
    a. have you ever used tools for architecture rule formalization and validation (in your projects)?
4. if yes, which one? (e.g. sotograph, sonarqube, archunit, jqassistant...)
    a. do you find them helpful?

Project Background

1. What is the size of the team?
    a. How many developers
    b. How many architects
2. What is the size of your software system you develop?
    a. Approximately How many lines of code?
    b. How many projects/components?
3. How is architecture documented in your projects? (wiki, word document, formal models, issue tracking system, plain text files, whiteboard, no documentation...)
4. are architecture rules documented in your project?
    a. if yes, how? separate (word/text/...) document, wiki, issue tracking system, no documentation
5. do you formalize architecture rules in your project?
6. do you use tools for validating architectural rules?
    a. if yes, which one? (e.g. sotograph, sonarqube, archunit, jqassistant...)
    b. if no: why not?

Questions regarding the CNL Formalization

For each rule, the following questions are asked:

1. How understandable is the formalization of the rule in CNL?
    a. perfectly understandable, largely understandable, partially understandable, largely not understandable, not understandable
    b. in case it is not understandable, why? what hinders the understandability?
2. How artificial/natural does the formalization appear to you?

    a. if it does appear artifical, why?
3. How well does the CNL formalization reflect the original intention of the architectural rule?
    a. are concepts and relations of the original rule well represented?
    b. if it does not reflect the rule well: why?
    c. how similar is the formalization with respect to the original one in natural language?

Overall Evaluation

1. In your opinion, how useful would it be having such a "natural" formalization of architecture rules that can also be validated?
2. would you use this approach for specifying and documenting architecture rules?
    a. yes: how would you use it?
    b. no: why not? what is missing? what are the disadvantages?
3. do you think developers would be more aware of the architecture rules when using this approach (because it is a rather intuitive description)?
4. do you think the CNL formalization can be potentially understood by all team members (architects and developers)?
5. do you think the approach would support developers and architects to respect the architecture rules during implementation?
6. what do you think are the key benefits of this approach?
7. what do you think are the key drawbacks/disadvantages/challenges of this approach?
8. which features should the approach provide additionally?
9. is there anything else you would like to say/discuss/add?