Example of a theory description (excerpt):

**Table 3** Constructs, propositions, example explanations and scope of the theory of UML-based development

| | |
|---|---|
| **Constructs** | |
| C1 | *UML-based development method* |
| C2 | *Costs* (total number of person hours in the project) |
| C3 | *Communication* (ease of discussing solutions within development teams and in reviews) |
| C4 | *Design* (perceived structural properties of the code) |
| C5 | *Documentation* (the documentation of the system for the purpose of passing reviews as well as for expected future maintainability) |
| C6 | *Testability* (more efficient development of test cases and better quality, i.e., better coverage) |
| C7 | *Training* (training in the UML-based method before the start of the project) |
| C8 | *Coordination* (of requirements and teams) |
| C9 | *Legacy code* (code that has not been reverse engineered to UML-models) |
| **Propositions** | |
| P1 | The use of a UML-based development method increases costs |
| P2 | The use of a UML-based development method positively affects communication |
| P3 | The use of a UML-based development method positively affects design |
| P4 | The use of a UML-based development method positively affects documentation |
| P5 | The use of a UML-based development method positively affects testability |
| P6 | The positive effects of UML-based development are reduced if training is not sufficient and adapted |
| P7 | The positive effects of UML-based development are reduced if there is insufficient coordination of modelling activities among distributed teams working on the same project |
| P8 | The positive effects of UML-based development are reduced if the activity includes modification of legacy code |
| **Explanations** | |
| E4 | The documentation is<br>– More complete<br>– More consistent due to traceability among models and between models and code<br>– More readable, and makes it easier to find specific information, due to a common format<br>– More understandable for non-technical people<br>– May be viewed from different perspectives due to different types of diagram |
| E5 | Test cases based on UML models<br>– Are easier to develop<br>– Can be developed earlier<br>– Are more complete<br>– Have a more a unified format<br>Moreover, traceability from requirements to code and test cases makes it is easier to identify which test cases must be run after an update |
| **Scope** | |
| | The theory is supposed to be applicable for distributed projects creating and modifying large, embedded, safety-critical subsystems, based on legacy code or new code |

Source:
Guide to Advanced Empirical Software Engineering
Shull, Forrest; Singer, Janice; Sjøberg, Dag I. K. (Eds.)
2008, XII, 388 p. 37 illus., Hardcover
ISBN: 978-1-84800-043-8