

Experience ND21 Situation Analysis Model

Purpose of the Experience situation analysis method ND21 is to help to estimate new development and enhancement projects. The model consists of 21 standard productivity factors. They are classified into four categories: Project, Process, Product and People. Each group except Product includes 5 factors. Each factor in each category has five alternative values. The basic idea in rating is that “the better the circumstances of the project are, the more positive rating the factor gets”.

“++” = Excellent situation, circumstances much better than in average case

“+” = Good situation, circumstances better than in average case

“+/-“ = Normal situation in the productivity point of view

“-“ = Bad situation, circumstances worse than in average case

“—“ = Very bad situation, circumstances much worse than in average case

Each alternative of each factor is weighted based on experience data. The ideal or target weights should be 1.10, 1.05, 1.00, 0.95 and 0.90 (from - - to ++) and they should be distributed normally, 5 - 20 - 50 - 20 - 5 %.

Definitions of Productivity Factors

Project	Involvement of the customer representatives
Project	Performance and availability of the development environment
Project	Availability of IT staff
Project	Number of stakeholders
Project	Pressure on schedule
Process	Impact of standards
Process	Impact of methods
Process	Impact of tools
Process	Level of change management
Process	Maturity of software development process
Product	Functionality requirements
Product	Reliability requirements
Product	Usability requirements
Product	Efficiency requirements
Product	Maintainability requirements
Product	Portability requirements
People	Analysis skills of staff
People	Application knowledge of staff
People	Tool skills of staff
People	Experience of project management
People	Team skills of the project team

1. Project factors 1.1 – 1.5

1.1 Involvement of the customer representatives; How actively the customer (can be interpreted as an end-user, purchaser or a defined market segment) participates in the development work.

— The customer is not interested in or has no time to participate requirements specification and the project in practice.

- Participation of the customer representatives is passive or rather formal. A small amount (less than 30 %) of the software functions is defined by them.

ND21

- +/- The customer representatives participate in the project with satisfactory activity. Approximately half of the functions (30-70 %) are defined and approved by them.
- + The customer representatives participate actively. The majority of the functions (over 70 %) is defined and approved by them (all the most important functions).
- ++ The customer representatives are available when needed. Most of the requirements are specified and controlled by them.

1.2 Performance and availability of the development environment; Performance level of the tool resources and equipment (tools, hardware, physical environment, network etc.) during the project.

- Shortcomings of the development environment are very common. Testing requires special arrangements because of operative production reserving all available resources.
- Shared equipment/machine resources. Delays in some work stages (for example compiling and testing).
- +/- Enough equipment/resources during the development life cycle. Reasonable workstations, software tools and physical sites for everybody.
- + Enough equipment and tool resources also for capacity peaks (efficiency, storage, response time etc. criteria fully met).
- ++ Dedicated, over-dimensioned development environments, in practice only for this project.

1.3 Availability of IT staff; The availability of the software personnel during the project.

- Great problems in availability of professional software personnel to most tasks (lots of customer and maintenance responsibilities simultaneously, special know-how required).
- The team members are involved in some other simultaneous projects (also maintenance responsibilities). Low priority for this job.
- +/- The key members of the project team are involved only in one other project at the most in addition to this project. That can have overload and has impact on this project.
- + The members of the project team are involved in this project almost full time. No special qualifications required. Some overload possible for key staff.
- ++ Qualified software personnel available when needed and they can participate fully on this project. Readiness for high overload for short times during the project.

1.4 Number of different stakeholders; Total number of involved organisations and parallel dependent projects, see value from the table below. Dependency between projects must be based on the deliveries, not on shared human resources.

Number of parallel dependent projects

Number of organisational units involved	>5	4-5	2-3	1	0
more than 3	—	—	-	+/-	+/-
2-3	-	+/-	+/-	+/-	+
only 1	+/-	+/-	+	++	++

1.5 Pressure on schedule; The pressure on schedule, goal schedule compared to first estimate of analogy.

- Goal schedule either extremely tight or very loose. The target duration either less than 50 % of an ordinary case or more than 100 % longer.
- Goal schedule clearly tight or clearly loose. The target duration either 50-80 % of an ordinary case or 50-100 % longer.
- +/- Goal schedule slightly loose, the target duration is about 20-50 % longer than an ordinary case.
- + Goal schedule at realistic level, the target duration less than 20 % longer than an ordinary case.
- ++ Goal schedule ambitious, moderately challenging, but not impossible. The target duration about 80-90 % of an ordinary case.

2. Process factors 2.1 – 2.5

2.1 Impact of standards; The quality of the existing standards and procedures applied in the project.

- The standards and basic practices must be followed strictly, but they are unstable and will be changing during the project.
- Standards are partially OK, more procedures must be developed for some tasks. Not familiar in the project or the organisation beforehand.
- +/- Well known but general standards that have been applied also before. Some tailoring needed for most major tasks.
- + Detailed standards, which have been applied in the same environment for some time.
- ++ Stable and detailed standards, which are already familiar for the team. Flexibility in application and good control of use.

2.2 Impact of methods; The use and quality of the methods to be exploited during the project.

- The project does not use any software engineering or management methods. Working based on traditional meetings, individual working.
- The use of methods is beginning, traditional concepts (structural analysis and design, top-down design...)
- +/- Generally known methods used (structural analysis and design, conceptual analysis, ER-modelling etc.)
- + Methods are integrated at detailed level and they cover most activities and team work, support exists.
- ++ Methods cover the whole lifecycle and are tailored to satisfy the specific needs of the project, well organised support.

2.3 Impact of tools; The use and quality of all kind of tools available for the project (programming, testing, documenting, project management etc.).

- Minimal set of basic tools, like editors, compilers and simple debugging tools.
- Basic tools, like interpreters, editors, compilers, debuggers, data bases, libraries.
- +/- Development environment, data base management system, support for most phases.
- + Modern tools, like CASE, project planning, application generators, testing tools. The standardised interfaces between the phases and/or tools.
- ++ Integrated CASE- environment, covers the whole lifecycle. All tools can support each other flexibly.

2.4 Level of change management; The stability and predictability of the functional requirements in the project, maturity of the change management process.

- Continuously new requirements. No real contract for the project. More than 30 % of the functions are (expected to be) new or modified compared to original requirements.
- Some of the accepted changes are essential and have an clear impact on the application architecture. Return to previous phases and modifications to previous deliveries. 15-30 % of the functions are new or modified.
- +/- Changes to specifications occur, but they can be managed and their impact is minor (less than 15 % new or modified functions).
- + Some changes to specifications, some new or adapted functions, some minor changes in data contents.
- ++ No new features during the project. For example a pure conversion project.

2.5 Maturity of software development process; Stability and conformance of the software development processes and lifecycle related activities in the project.

- Requirements volatility cause return to previous phases many times during the project. No means to eliminate that behaviour. No conformance with quality models.
- Return to previous phase may happen many times in different parts of the software. Basic practices conform mostly with internal procedures.
- +/- Some rework caused by continuous changes and late discovered defects. Some integration problems is expected during testing and release processes. Conformance with internal audits.
- + Stable progress, no return to previous steps, reviews find most of the defects early. Conformance with quality models.
- ++ Complete straightforward progress in all parts of the software, no defect or integrity problems expected. Strong tool support for all processes. Strong conformance with quality models.

3. Product factors 3.1 – 3.6

3.1 Functionality requirements of software; Compatibility with end-user needs, complexity of the requirements, level of integration.

- Virginal and complex application area, security critical big (thousands of fp's) multi-tier system for various, multi-cultural users.
- New , interoperable application area with some complex features, demanding special understanding from the users and developers.
- +/- Partially automatised, integrated application area and a middle size (600-1000 fp's) application with normal security requirements.
- + A largely automatised application area and the application with less than 5 interfaces with other systems, no specific security requirements.
- ++ Very mature, straightforward and easy application area, a small (less than 200 fp's) stand-alone application for a small group of users.

3.2 Reliability requirements of software; Maturity, fault tolerance and recoverability in different types of use cases.

- Operation faults may endanger human lives or cause great economic or environmental losses, the application must recover without losing any data in any case.
- The software is **either** a part of a large integrated real-time system, where all operation faults will cause problems to many other applications and /or thousands of end-users **or** a part of broadly delivered consumer product with extremely high maintenance costs.
- +/- Not longer than 2 hours breaks acceptable, but the system level recovery routines will be adequate. Operation faults do never cause remarkable economic or image losses or any danger to human beings.
- + Need for operation not continuous but daily. Time for recovery typically 24 hours and even the worst operation faults will cause losses of not more than hundreds of euros (or dollars).
- ++ Need for operation periodic. Break of a couple of days will cause no harm to the end-user organisation. Typically an independent stand-alone back-office software for only a small number of users.

3.3 Usability requirements of software; Understandability and easiness to learn of the user interface and the logic of the work-flow.

- A very big number of different types of end-users all over the world, with different levels of experience at software usage, a high-level customisation and help facilities required.
- 2-3 different types of users with various skills, requiring automatised multi-level help function, the use of software during interactive customer service.
- +/- A big number of end-users with equal skills, typical simple help function adequate, calm and stressless circumstances for the use of software.
- + Not more than tens or hundreds of homogenous users at maybe more than one location, literal instructions supported by a very simple help function adequate, no special care for the operability.
- ++ Only few users, all located at one site, not very frequent use, direct help and support from the developers

easy to arrange when needed.

3.4 Efficiency requirements of software; effective use of resources and appropriate performance in every use case and under any reasonable work-load.

- Complex database with millions of data records and transactions per day, thousands of simultaneous end-users with various data requests, almost all response times critical, parallel batch and online processing, operation close to the capacity limits during rush-hours.
- Big database, hundreds of simultaneous end-users, most of response time requirements critical, alternating batch and online processing needs, narrow shifts for batch processing.
- +/- Big database, less than million data records and less than one hundred simultaneous end-users, response time requirements are flexible, wide shift for batch processing.
- + Average database by volume and structure, straightforward and predictable data requests from few simultaneous end-users.
- ++ Simple and small database, no simultaneous end-users or complex data requests, total number of transactions not more than tens per day.

3.5 Maintainability requirements of software; Stability of the environment, target lifetime of the application, criticality of defect diagnostics and test performance.

- Very large strategical (target lifetime more than 20 years) software at a volatile business area with frequent changes of laws and standards and business rules. Also the maintenance speed is essential, logging and the defect messages must be clear, exact and guiding the developers.
- Large software (target lifetime from 10 to 20 years), frequent changes of laws or standards or business practices. Time to analyse defect messages, change the programs and test them is always some hours but not more.
- +/- Average size tactical (target lifetime from 5 to 10 years) software, monthly changes of laws, standards and business practices. Maintenance timing is reasonably flexible, a couple of days rather than hours, an application specific error log needed.
- + Rather small rarely changing software (target lifetime from 2 to 5 years), no application specific diagnostics needed.
- ++ Temporary software (target lifetime less than 2 years) with no connections to changing laws, standards or business practices.

3.6 Portability requirements of software; adaptability and installability to different environments, openness of architecture and structural components.

- Users of the software are located in many kind of organisations, with various platforms (hardware, browsers, operating systems, middleware, data communication protocols etc), various versions and various upgrading frequencies.
- The software must operate on a couple of different platforms (hardware, browsers, operating systems, middleware, data communication protocols etc) and on several versions of them.
- +/- Every version of the software must run on several versions of a certain platform (hardware, browser, operating system, middleware, data communication protocol etc), the upgrading frequencies of the users are rather predictable.
- + The software must run on a certain platform (hardware, browser, operating system, middleware, data communication protocol etc), but the use of system level services is limited because the upgrading process is only partially manageable.
- ++ The software must run on a certain platform (hardware, browser, operating system, middleware, data communication protocol etc) which upgrading process is completely manageable (for example most of the mainframe environments).

4. People factors 4.1 – 4.5

4.1 Analysis skills of staff; The analysis skills of the project staff in the beginning.

- No experience in requirements analysis and from the similar projects.
 - Part of the staff (less than 30 %) has experience on the analysis and design activities in similar projects.
 - +/- 30-70 % from the project staff has experience on analysis work. The project has also an experienced member.
 - + Most people have experience on specifications and analysis, the project manager is professional in analysis work.
 - ++ The project staff consists of first-class professionals, strong vision and experience on requirements analysis.
- 4.2 Application knowledge of staff; Knowledge of the application domain in the project team (both the supplier and the customer) at the kick-off moment.

- The application domain experience in the team is very small, less than 6 months in average.
- The application experience is small, some members of the project staff have application experience, from 6 to 12 months in average.
- +/- The team has quite good experience of working at the application area, 1-3 years in average.
- + The application experience is good both on the supplier and the customer sides. Also business dynamics is known. The experience is 3-6 years in average.
- ++ Both the supplier and the customer representatives know the application area very well, including the understanding of the business as total. The average experience is more than 6 years.

4.3 Tool skills of staff; Average experience of the project team (supplier, customer) on development and documentation tools at the kick-off moment. All important tools must be included here.

- The team has no experience of the necessary tools. The average experience time is less than 3 months.
- The tools experience is less than average, some members have experience of some tools, the average is from 3 to 6 months.
- +/- The tools experience is rather good. Some members of the team know the most important tools well. The experience from 6 to 18 months in average.
- + Most of the team members know well the tools needed in the project. Some members can give support in the use. Experience time from 18 months to 3 years in average.
- ++ The team knows all the tools very well. Support available for the specific needs of the project. The experience more than 3 years in average.

4.4 Experience of project management; The experience of the project manager and other key staff.

- Project manager has no previous experience of similar type of projects, not even as a member.
- Project manager has experience of smaller similar type of projects, as a member or partially responsible of management.
- +/- Project manager has experience of managing at least one similar project.
- + Project manager is experienced of several similar projects, some success also on difficult projects.
- ++ Project manager is a real professional with experience of managing many kind of software projects.

4.5 Team skills of project team; The ability of the project team members to work effectively together following the best project practices.

- Scattered group of people, no experience of working as a member of a project team, minimal project and collaboration skills.
- Some of the team members have previous experience of similar projects, no experience of working together with each other, but some motivation to try exists.
- +/- Most of the team members have experience of similar projects. The commitment on the project goals and the

ND21

motivation to work together is satisfactory.

+ The project group is active, the team members have experience of working together with each other and all of them are able to share responsibilities rather effectively.

++ The team can solve most of the difficult problems together, using innovative and inspiring group work. Superior team spirit.