SIOUX TECHNOLOGIES

Finetech GmbH & Co. KG
Dr. Sylvio Schneider
Team Lead Software development
Boxberger Str. 14
12681 Berlin
Germany

| | |
|---|---|
| **Our Reference** | 22-09-014-KG |
| **Subject** | Quotation Finetech SW Consulting |
| **Cc** | - |
| **Appendices** | 1 Architecture Questionnaire, 2 Non-Functional Requirements, 3 Estimation, 4 Quotation, 5 RFQ provided by Finetech |
| **Date** | 23 September 2022 |

Dear Dr. Schneider,

Sioux Technologies GmbH ("Sioux") is pleased to have been given the opportunity to submit a proposal to assist Finetech GmbH & Co. KG with the execution of the software consulting with the goal to analyse the current system and software stack to be able to evaluate possible solutions how to make the software stack fit for future needs.

With this quotation Sioux is providing Finetech with a plan how to realize the "Finetech SW Consulting" request formulated in Project-Memo 'Anfrage für ein Beratungsprojekt zur Verbesserung des Softwaresystems und -entwicklung' of 27-07-2022 (attached in *Appendix 5*). Based on the information provided in this document, Finetech is able to judge thoroughness, completeness and fairness of our proposal. Estimation and Quotation can be found in *Appendix 3: Estimation* and *Appendix 4: Quotation*.

Sioux' general terms and conditions of sale are applicable to this quotation. To the extent of any conflict between this quotation and the general terms and conditions of sale, this quotation shall prevail.

With kind regards,

Sioux Technologies GmbH

Klaus Gruber
Development Manager

Sioux Technologies GmbH
Am Weichselgarten 36
D-91058 Erlangen
Germany
T +49 9131 812928 0
www.sioux.eu
info_de@sioux.eu
Geschäftsführer: Arnoud de Geus

ICC

## Sioux Approach

Sioux suggests splitting the SW consulting in the following main work packages:

1. Assessment of the System Architecture, Software Architecture and Technology
2. Assessment of the Software Process and Organisation
3. Compilation of SW key requirements
4. Development of possible solution scenarios
5. Evaluation of possible solution scenarios

## System Architecture, Software Architecture and Technology

Sioux will investigate regarding the scope:

- Status of current products, their context and users
- Roadmap of Finetech, with regards to product development and keys to success
- Overall Software Architecture (layer, approach)
- Programming languages
- Operating Systems
- Software layers to Motion and Vision
- Used technologies of main components
    - Data handling
    - GUI
    - Vision handling
    - Application
    - Host interfaces
    - 3rd party SW
- Extendibility and maintainability, complexity of the code base
- Legacy code, supported configurations, variants

Sioux will also assess the tool chain used during software development.

**Sioux suggested approach:**
The assessment is a series of interviews, combined with a high-level investigation of actual software items such as documentation, source code, tooling.

**Assessment preparation**
Get a basic understanding of the products of Finetech, understand the product roadmap, understand on a high level the software stack and technologies used. Preparation is needed to get a view of the architectures at the higher level prior to the interviewing process, e.g., Architecture and Interface documentation.

**Activities**

- Review of architecture
    - Programming Languages
    - Operating systems
    - Software Layering
- Review of configuration management system
- Review of the complexity of the code base
- Review the Software Development Environment
- Testing procedures
- Deployment tools

**Interviews**
Sioux will be conducting interviews with the key stakeholders to assess the technologies and technology stack used and to understand more about the considerations why these architectures are chosen, and what the issues are with the architectures according to the Finetech team members.

ICC

As input for the interviews, we will not use a formal architecture assessment's model / approach but a practical though structured approach (to keep the effort in this phase moderate), see for reference *Appendix 1: Architecture Questionnaire*. We used this questionnaire before to collect in a short period valuable feedback. Of course the spirit of more specific and more formal architectural models and procedures will be used such as:

- ATAM: Method for Architecture Evaluation
  https://resources.sei.cmu.edu/asset_files/TechnicalReport/2000_005_001_13706.pdf
- CAFCR: A Multiview Method for Embedded Systems Architecting
  - https://www.gaudisite.nl/ThesisBook.pdf
  - https://www.gaudisite.nl/ArchitecturalReasoningBook.pdf
  - https://esi.nl/academy/what-we-offer/learn/systems-architecting-what-we-offer/architecting-for-business-value

Only moderate focus will be given on the non-functional requirements, from the Q-model[1] (See *Appendix 2: Non-Functional Requirements of the Q-Model*). Only focus on Extendibility, Maintainability and the complexity of the architecture will be analysed.

**Analysis & Benchmark**

The collected information will be benchmarked against our common understanding of comparable software architectures for comparable systems. All the items will have the main focus of analysis & benchmarking. The status of the benchmark and proposed improvements will be structured on the same list.

During the assessment Sioux will estimate, if any current state-of-the-art approaches for architecture improvement methods for similar complex machines can be applied to the Finetech platforms, such as:

- Model extraction for software legacy
  https://esi.nl/research/output/methods/model-extraction-for-software-legacy
- Model-based software analysis
  https://esi.nl/research/output/methods/model-based-software-analysis
- Model-based software transformation
  https://esi.nl/research/output/methods/model-based-software-transformation

The goal of Sioux will be to advise how Finetech can improve their software platforms. Therefore evolutionary scenarios and also revolutionary (and complete new greenfield) scenarios will be evaluated.

[1] Non-functionals usually have a mayor impact on software architecture patterns

SIOUX TECHNOLOGIES

**Reporting & Presentation**

Sioux will be delivering the results of this investigation in the form of a powerpoint presentation, which will be also presented to Finetech.

## Software Process and Organisation:

Sioux will be investigating the Software processes and the Software Organisation of Finetech.

**Scope:**

- Software Development process
- Software testing process and releasing process
- Adherence to these processes
- Involved roles and interfaces
- Effort estimation (accuracy, structure, process)
- Organisation (global SW and interfaced departments)
- Bugrate, needed effort for features, bugfixes
- Software Requirements and specifications
- Software documentation, release notes

**Sioux suggested approach**
In order to get an understanding of the Software Process and the Software Organization, Sioux proposes to start with a review of the available documentation about the Software Development Process, Quality Management Systems and other relevant information. We will start with a broad scope and let ourselves be guided by the first interview results to determine the topics which matter most.
The Software Process and Organisation assessment will be done on a practical level.

**Assessment preparation**
Preparation is needed to get a view of the intended processes, i.e., the Quality Management System(s), if available. Think of structure, topics, supporting material. Sioux will be sending, as preparation for the interviews, a list with requested documentation. Sioux will be analysing the provided information prior to the interviews.

**Activities**

- Review of Quality Management System(S)
- First interviews
- Review of the Software Testing Procedures
- Review of the Software Documentation

**Interviews**
Sioux will be conducting interviews with the key stakeholders to assess the process and organisational related aspects of the Finetech SW development. During the interviews an open atmosphere is required. Therefore, the interviews are supposed to be confidential unless interviewees don't mind being referred to. Sioux will use the interviews to first of all understand the processes and secondly to check or the processes are being followed.

ICC

**Analysis & Benchmark**
Before presenting interview results, the outcome will be structured along the reference model topics (in Powerpoint). In addition, a 'score sheet' in Excel listing more details per topic will be prepared.

Although the assessment is no theoretical or full-fledged CMMI audit, the CMMI model does provide a checklist for the topics as addressed in the project briefing, sector 2.2 as attached in Appendix 5. Since the CMMI framework is not covering all the topics, in addition we will be using the ISO12207 standard to assess the remaining topics. See for more information about the chosen models:

- https://www.cmmiinstitute.com/
- https://www.iso.org/standard/63712.html

With respect to the issue of 'Bugrate, needed effort for features, bugfixes', this point will be given special attention. How are CR's/PR's (Issues) managed? How are (especially Field Problem Reports) injected into the organisation and how are they managed to successful closure and delivery to end customer? How does this influence the 'planned' roadmap and release planning of the Finetech R&D organisation? How are Issues administered, tracked, and estimated? What is the quality of the Issue estimation, and how are they injected to a SW release schedule? How many issues are open, is this increasing or decreasing? What is the characterisation of the issues: What percentage is more lack of functionality, single end customer specific wishes, real issues of functional non-compliancy and what percentage is non-robust SW behaviour?

The organisational aspects can be approached using experience, common-sense and management literature in mind. Topics should be governance model, structure, communication, reporting lines, distribution of locations, etc.

**Reporting & Presentation**
For reporting use will be made of the same checklist structure to indicate highlights and lowlights and opportunities for improvement. Looking at the list of stakeholders we can imagine one-size-fits-all feedback will not work and we would think of slide sets tailored to specific audiences at different occasions, which will be also presented to Finetech.



IC

## Compilation of SW key requirements:

Based on the assessment Sioux will compile the key requirements for the software stack. Together with the stack holder from Finetech these key requirements will be reviewed to ensure, that all key requirements are within focus.

## Generation of possible solutions for SW stack:

Based on the assessment and the key requirements Sioux will develop and document possible solution scenarios for the SW stack.
There will be at least one scenario for evolutionary approach and at least one scenario for revolutionary approach.
For these possible scenarios a rough architecture will be defined and documented.

## Evaluate possible solutions for SW stack:

Together with Finetech the evaluation criteria will be defined and weighted.
Potential criteria are:

- Future proof
- Expandability
- Testability
- Stability
- ...

Based on the criteria the possible solutions will be evaluated and the results will be documented.
This will be the base for the major decision to be done regarding the right way for Finetech regarding the future SW stack.

### Reporting & Presentation

Sioux will be delivering the results of this investigation in the form of Word documentation including the details (SW key requirements, possible solutions, weighted evaluation criteria and evaluation results of possible solutions) and a Powerpoint presentation.



ICG

SIOUX TECHNOLOGIES

## Appendix 1: Architecture Questionnaire:

This is a checklist to assess a body of software. The purpose might be to acquire it, integrate with, do maintenance for or be part of some trade-off what software stack to start from. Here is a list of questions for each main aspect or angle to assess the software.

**Main aspects:**

1. Architecture
2. Documentation
3. Development Environment
4. Quality
5. Security
6. Technology
7. Licenses

**Architecture:**

1. Is it clear which needs this architecture is supposed to satisfy?
2. Are there any pictures, and are these (still) accurate? Any design patterns used?
3. How many different configurations are supported by the software?
4. Are there APIs? Versioning of these?
5. Are there dependencies between components, including 3rd party?
    a. e.g. entangled mesh of inter-dependent responsibilities, or clear separation of independent components with well-defined interfaces
6. What are the scalability. costs of more users, and cost of more functionality added?
7. Are there a bad weather strategy / requirements formulated (FMEAs, HAZOPs, etc)
8. Is there a model driven approach? Supermodels, MPS, Protobuf/GRPC, OpenAPI? How about state / workflow?
9. Are there simulators, and how well do they represent the production environment?
10. Are there 3rd party frameworks used (support, availability)?
11. Is there a product roadmap available?
12. Is it documented what the product variation points are? (i.e. how many variations of the product does the software support?)
13. How large is the software in terms of KLOC (per component, total).
14. Is it possible to determine the amount of effort required to create the software, how large were the development teams?

**Documentation:**

Is there any documentation, and how accurate is it? English or another language?

1. Are Use-cases documented?
2. What is the format of the documentation? Office / Web / In code (generated) / other?
3. Are designs present and clear
4. Are requirements present and clear?
5. Test plans and reports present and match the requirements?
6. Any standards or norms applicable?
7. Is Traceability needed, and is it present and accurate
8. SOUP (software of unknown providence) needed?



IC G

**Development Environment:**

1. Is it clear (documented) how to set the project / services up locally for a new developer? Is the environment deterministic and reproducable?
    a. What IDEs, Tools, Development Languages and versions of these are used?
    b. What build processes and outside dependencies are required?
    c. Are static code analysis tools used (formatting, linting, …)
    d. Could I get it to run without outside help?
2. Are there Unit Tests available?
3. Is the process of shipping to production present? (e.g. is there an installer present?)
4. Is there CI/CD set up? How? Documentation?
5. How is the daily work of a developer organized?
    a. Issue and bug tracking (e.g. Jira, GitHub, Azure Devops, etc)?
    b. Is the development flow documented? (e.g. git flow, merge checks)

**Quality:**

1. Code quality SLOC
2. Cyclomatic complexity: Issues reported by tools (Sonaqube, etc)
3. Test Approach
    a. Test framework? Automated unit vs System tests; Performance tests
    b. Test coverage?
4. Versioning of components, interfaces, config files, machine parameters/calibrations.
5. Maintainability → derived from coupling cohesion, tested, code quality/complexity; known technical debt
6. Is there a list of backlog or still open issues (bugs)?

**Security:**

1. Data security (backups / restores and level of automation)
    a. Is there any sensitive data? (GDPR?)
2. How are (access) keys managed? Key vault?
3. Multitenancy? How is data kept separate?
4. Are penetration tests available?
5. Was there a security audit (e.g. vulnerabilities found (and fixed) by tools)

ICC

**Technology:**

1. What technology is used in each component? (e.g. front-end / back-end)
2. What is the target system for deployment? (e.g. web technology for mobile)
3. Are there any trade-offs documented for the choices?

**Licenses:**

1. Open source used?
2. List of individual licenses available?
3. License cost?
4. Does the end user need to sign?

SIOUX TECHNOLOGIES

## Appendix 2: Non-Functional Requirements of the Q-Model

| CATEGORY | Q - FACTOR | DESCRIPTION | L | M | H |
|---|---|---|---|---|---|
| **Creatability** | Ease of creation | Degree of effort required to create the system according to the stated requirements | | | |
| | Outsourceability | Degree to which the implementation of parts of the system can be outsourced | | | |
| | Buy-in | Degree to which existing components can be applied in the system | | | |
| | Conformance | Adherence to technology or industrial standards for product (process) | | | |
| | Manufacturability | To be manufactured with low cost, high throughput, low drop out, etc. | | | |
| | Environment impact | Regarding manufacturing, life time, recycling costs, disposable consumption, power consumption | | | |
| **Functionality** | Suitability | Providing an appropriate set of functions for specified tasks and user objectives | | | |
| | Interoperability | Interaction with one or more specified systems | | | |
| | Security | Detection and prevention of unauthorized access (accidental or deliberate) to programs or data | | | |
| | Compliance | Adherence to application related standards or conventions or regulations to laws | | | |
| | Integrability | Degree to which components of the system can be easily integrated | | | |
| | Configurability | Adaptation the system to different needs | | | |
| | Compatibility | Of the system with earlier or future systems | | | |
| **Reliability** | Correctness | Degree to which the system conforms to the stated requirements | | | |
| | Accuracy | Providing the right or agreed results or effects (including data with needed degree of precision) | | | |
| | Availability | Degree to which the system is available to the user on the time it is needed | | | |
| | Fault tolerance | Maintaining a specified level of performance in case of system failures or infringement of its interface | | | |
| | Recoverability | Re-establishing the level of performance and recover data directly affected in the case of a failure | | | |
| | Safety | Absence of unsafe system conditions that could lead to loss of life or liability, or damage to property | | | |

| CATEGORY | Q - FACTOR | DESCRIPTION | L | M | H |
|---|---|---|---|---|---|
| **Usability** | Understandability | Enabling the user to understand whether the system is suitable, and how it can be used for particular tasks | | | |
| | Learnability | Enabling the user to learn the system's application | | | |
| | Operability | Enabling the user to operate and control the system (required effort) | | | |
| | Explicitness | Clarity of the system with regard to its status | | | |
| | Responsiveness | Of the system regarding reaction according to user expectations (feedback during processing) | | | |
| | Customisability | Enabling the system to be customised by the user to reduce effort needed for use and increase satisfaction | | | |
| | Clarity | Clarity of making the user aware of the functions the system can perform | | | |
| | Helpfulness | Availability of instructions for the user on how to interact with the system | | | |
| | Attractiveness | To be liked by the user | | | |
| **Efficiency** | Time behaviour | Providing appropriate response/processing times and throughput rates (no degeneration over time) | | | |
| | Resource utilisation | Using appropriate resources in an appropriate time when performing the functions (memory, comm.) | | | |
| | Resource utilisation | Using appropriate resources in an appropriate time when performing the functions (memory, comm.) | | | |
| **Maintainability** | Analysability | To be diagnosed for deficiencies or causes of failures, or for the parts to be modified to be identified | | | |
| | Correctability | Enabling an identified fault to be removed | | | |
| | Expandability | Increasing the system's functionality or performance to meet new needs | | | |
| | Stability | Minimising unexpected effects from modifications of the system | | | |
| | Testability | Enabling the developed or modified system to be validated | | | |
| | Scalability | Supporting modifications that strongly increase the system's internal capacity (same functionality) | | | |
| | Serviceability | Servicing the system in its operating environment (ease, effort) | | | |

| CATEGORY | Q - FACTOR | DESCRIPTION | L | M | H |
|---|---|---|---|---|---|
| **Portability** | Adaptability | To be modified for different specified environments (including hardware/software independence) | | | |
| | Co-existence | Co-existing with other independent software in a common environment sharing common resources | | | |
| | Installability | Of the system to be initially installed, set up, calibrated, etc. in a specified environment | | | |
| | Upgradability | To be upgraded (with new functions, releases, etc.) in the system's operating environment | | | |
| | Replaceability | To be used in the place of specified other system (parts) in the environment of that system | | | |
| | Reusability | To be complete or partially reused in another system | | | |

## Appendix 3: Estimation

| | | Project Lead | Process Consultant | System Architect | Software Architect | Software Designer |
|---|---|---|---|---|---|---|
| | **Total (hours)** | 87 | 32 | 167 | 368 | 260 |
| **Preparation Assessment** | | 10 | 4 | 16 | 16 | 16 |
| **Interviews** | | | | | | |
| | Managing Director | 2 | | 2 | | |
| | Product Manager | 2 | | 2 | | |
| | Development Manager | 2 | | 2 | | |
| | SW Department Manager | 2 | | | 2 | |
| | Electronics Department Manager | 2 | | 2 | | |
| | Mechatronics Department Manager | 2 | | 2 | | |
| | SW Teamlead | | | 2 | 2 | |
| | SW engineer | | | | 2 | 2 |
| | SW engineer | | | | 2 | 2 |
| **Work out interviews** | | | | | | |
| **Architecture** | | | | | | |
| | Generate overview of System Architecture | | | 25 | 10 | |
| | Generate overview SW Architecture (layer, approach) | | | | 45 | 20 |
| | Programming languages | | | | 2 | 2 |
| | Operating Systems | | | | 2 | 2 |
| | Software layers to Motion and Vision | | | | 8 | 2 |
| | Used technologies of main components | | | | 2 | 2 |
| | Data handling | | | | 2 | 2 |
| | GUI | | | | 2 | 2 |
| | Vision handling | | | | 2 | 2 |
| | Application | | | | 2 | 2 |
| | Host interfaces | | | | 2 | 2 |
| | 3rd party SW | | | | 2 | 2 |
| | Extendablility and maintainability, complexity | | | | 8 | |
| | Legacy code, supported configurations, variants | | | | 6 | 20 |
| **Tools** | | | | | | |
| | Sourcecode repository | | | | | 1 |
| | Binary repository | | | | | 1 |
| | Documentation tools | | | | | 1 |
| | Ticket system, scrum tools | | | | | 1 |
| | Release Management tools | | | | | 1 |
| | Test Tools / Systems | | | | | 1 |
| | Deployment tools | | | | | 1 |
| | Remote support tools | | | | | 1 |
| **Processes** | | | | | | |
| | Software Development process itself | | 4 | | 2 | 8 |
| | Software testing process and releasing process itself | | 2 | | 2 | 8 |
| | Adherence to this process | | | | | 8 |
| | Involved roles and interfaces | | 2 | | | 8 |
| | Skillsets of individuals related to their roles | | 6 | | | 4 |
| | Effort estimation (accuracy, structure, process) | | 4 | | | 8 |
| | Organisation (global SW and interfaced departments) | | 2 | | | 4 |
| | Bugrate, needed effort for features, bugfixes | | | | | 4 |
| | Software Requirements and specifications | | | | 4 | 4 |
| | Software documentation, release notes | | | | 4 | 8 |
| **Reporting** | | | | | | |
| | Written summary and overview of the issues, topics from the structured interviews | 5 | 5 | 20 | 20 | 8 |
| | Written report of the state-of-the-art benchmark/comparison | | | 16 | 16 | 16 |
| | Written report of identified gaps | | | 16 | 16 | 16 |
| **Requirements** | | | | | | |
| | Compilation of the SW key requirements | 4 | | 8 | 20 | 4 |
| **Possible Solutions** | | | | | | |
| | Development and documentation of possible solution szenarios for SW stack | | | | 45 | 20 |
| | Define and document rough architectures for possible szenarios (at least one evolutionary and one revolutionary approach) | | | 20 | 80 | 20 |
| **Evaluation of Solutions** | | | | | | |
| | Define evaluation criteria | 4 | | 4 | 4 | |
| | Evaluation of possible scenarios based on evaluation criteria and documentation. | 15 | | 14 | 20 | 8 |
| **Reporting** | | | | | | |
| | Progress reporting (management update, steering committee meetings) | 37 | 3 | 16 | 16 | 16 |

## Appendix 4: Quotation

This offer is based on fixed prices and times.

All prices are net prices, plus VAT, ex works Sioux Erlangen.

The estimated effort of this project is 114 person days. This assessment is based on the conversations and information that were available at the time of creation of this offer. Together with Finetech, Sioux strives to realize all activities within the above-mentioned project duration. Both parties agree that the actual effort may differ from the estimated forecast. If an increased effort is required, Sioux will announce this to Finetech and will be discussed and agreed upon together before additional activities will be performed.

| grade | name | role | FTE | Nov '22 work hrs | Dec '22 work hrs | Jan '23 work hrs | |
|---|---|---|---|---|---|---|---|
| 8 | NN | Project Management | 20% | 32 | 29 | 26 | 87 |
| 8 | NN | Process Consultant | 10% | 16 | 16 | 0 | 32 |
| 8 | NN | System Architect | 35% | 62 | 56 | 49 | 167 |
| 7 | NN | Software Architect | 70% | 123 | 112 | 132 | 367 |
| 5 | NN | Software Designer | 50% | 88 | 80 | 92 | 260 |
| | | | | 0 | 0 | | 0 |
| | | | | | | | Total |
| **Total Hours** | | | 1,9 FTE | 321 | 293 | 299 | 913 |
| **Total Hours x Uniform project rate** | | | € 125,00 | € 40.100 | € 36.600 | € 37.375 | € 114.075 |
| **Costs for Travel and Lodging** | | | | € 2.500 | € 2.500 | € 1.250 | € 6.250 |
| **Costs for Prototyping PCB's, EMC testing, etc** | | | | € 0 | € 0 | € 0 | € 0 |
| **Costs for Software Licensing** | | | | € 0 | € 0 | € 0 | € 0 |
| **Unforeseen costs** | | | 5% | € 2.130 | € 1.955 | € 1.931 | € 6.016 |
| | | | | **€ 44.730** | **€ 41.055** | **€ 40.556** | **€ 126.341** |

This results in a total price of: € 126.341,00 plus total Vat.

Project management and warranty claims are included in the cost estimates.
A key to success of this project is willingness and availability of the Finetech team to support Sioux in the assessment. The needed roles from Finetech team are listed in "appendix 3".
The quality of the outcome of the assessment is directly dependent on the cooperation.

Payment conditions

- € 40.000,00 Interviews done (30.11.2022)
- € 50.000,00 Report`s to assessment of the System Architecture, Software Architecture and Technology, Software Process and Organization done and presented (31.12.2022)
- € 36.341,00 Evaluation report of possible solutions done and presented (31.01.2023)

Validity of the offer
We are committed to our offer until 14.10.2022



ICG

## Appendix 5: RFQ, provided by Finetech:



KC

finetech

# Anfrage

**für ein Beratungsprojekt zur Verbesserung des Softwaresystems und -entwicklung**

**Finetech GmbH & Co. KG**
**Boxberger Str. 14 | 12681 Berlin**
www.finetech.de


**Dr. Sylvio Schneider**

- Gruppenleiter Softwareentwicklung und Projektleiter des Vorhabens

Phone: +49 (30) 936681-563
Mail sylvio.schneider@finetech.de


1( C-

1. Kurze Informationen zum Unternehmen Finetech ..... 3

1.1. Allgemeine Unternehmensinformationen ..... 3

1.2. Überblick Produktion ..... 3

2. Projektrelevanter Betrachtungsgegenstand ..... 4

3. Ziele eines Beratungsprojekts ..... 4

3.1. Übergeordnetes Projektziel ..... 4

3.2. Teilziele ..... 5

3.3. Sonstige Projektanforderungen ..... 5

ICG

# 1. Kurze Informationen zum Unternehmen Finetech

## 1.1. Allgemeine Unternehmensinformationen

- Umsatz: 25 Mio. €, Mitarbeiteranzahl: 180
- Produkte:
  - Maschinen für die Mikromontage
  - Manuelle, halb- und vollautomatische Die-Bonder für höchstgenaues Platzieren, Montieren und Packaging
- Anwendungen:
  - Bonden von Laser Bars, Dioden, VCSEL/PD-Montage
  - mehrstufige Aufbau opto-elektromechanischer Baugruppen (MEMS/MOEMS) für Produkte z. B. der Kommunikationsbranche und der Medizintechnik.

## 1.2. Überblick Softwareentwicklung

- ca. 6 Mitarbeiter
- Aufgaben: Tagesgeschäft, wie Bugs und Verbesserungen, sowie Integrations- und Entwicklungsprojekte
- Programmierung der Maschinensteuerung und –bedienung in Delphi
- Mikrocontroller-Programmierung in C++
- Hardwareintegrationen mittels Hersteller-DLLs
- Schnittstellenprogrammierung: CAN-Bus, EtherCat, Ethernet, USB3


# 2. Projektrelevanter Betrachtungsgegenstand

Software, wie z.B. Struktur, Architektur, Entwicklungsprozess, Entwicklungsumgebung

- Die Systeme bestehen aus Hardware für das Ausführen, sowie einem Windows-Rechner mit einer komplexen Steuersoftware
- Die Steuersoftware ist in Delphi als Monolith geschrieben und umfasst ca. 7 Mio LoC für das User-Interface sowie dem gesamten Steuerungsablauf (1,1 Mio. LoC Eigenentwicklung)
- Versionierung der Software erfolgt mit SVN
- Die Schnittstellen zur Hardware sind CAN-Bus und Ethernet
- Die Systeme sind autark und haben keine Möglichkeit von außen gesteuert zu werden oder Daten an eine zentrale Infrastruktur zu senden; MK: TeamViewer; MBo: Updates auf Nachfrage, von außen gesteuert
- Es existiert keine Integration in aktuelle Produktionssteuer-Systeme (SCADA, MES); MK: ProcessExchange, MES-Ansätze vorhanden und SECS/GEM

Die Software zeichnet sich dadurch aus, dass Maschinenfunktionen vom Kunden parametrisiert werden können und somit Prozesse (die beim Wettbewerb durch Programmierung umgesetzt werden) vom Kunden angepasst oder mit vertretbarem Aufwand geändert werden können.

Software Delphi:

- Kann nur mit Windows Betriebssystem arbeiten (Linux-Compiler prinzipiell möglich)
- Nutzung REST und gRPC möglich
- Verwendung HTTPS möglich mit Zertifikat
- Verwendung MQTT (Schnittstelle für IoT) geht nicht (laut Roadmap geplant)
- Immer wieder Unsicherheit über die Zukunft der Sprache
- Finden nur sehr schwer neue Mitarbeiter

# 3. Ziele eines Beratungsprojekts

## 3.1. Übergeordnetes Projektziel

Die bestehende Lösung soll auf nachfolgende Anforderungen überprüft werden

- Die Maschinen müssen in vernetzten Umgebungen eingebunden werden können
- Maschine2Maschine Kommunikation muss möglich sein
- Aus Wartungsgründen muss man auf die Maschinen von außen zugreifen können bzw. bei vorausschauender Wartung kontinuierlich Daten überwachen
- Maschine kann über Tablet angesprochen
- Die Lösungen müssen weltweit funktionieren
- Mit den entstehenden Daten müssen neue Services machbar sein
- Die Architektur muss zukunftsfähig sein. Dies gilt auch für die korrespondierenden Programmiersprachen
- Die Produktivität der Software-Entwicklung muss „state of the art „ sein

und Lösungen mit Handlungsempfehlungen erstellen werden.
(Aufwandsabschätzung, Kostenplanung, Ressourcenplanung, Risikobetrachtung, Zeitplan & Erfolgswahrscheinlichkeit)

## 3.2. Teilziele

- IST-und Potentialanalyse der Software und Softwareentwicklung
- Konkrete Aussage, welche Vorgehensweise zur Optimierung der Software/-Entwicklung (und ggf. angrenzender Bereiche) vorgeschlagen wird und wie Sie zur Verbesserung unterstützen könnten
- Vorschlag konkreter Quick Wins-Verbesserungen (sichtbare Verbesserung), die unmittelbaren Nutzen haben und nach Projektbeginn schnell geliefert werden können
- Konkrete Aussage zum Potenzial hinsichtlich Steigerung der gesamten Produktivität im Rahmen eines Beratungsprojektes (Laufzeit 3 Monate).
- Best Practice im Maschinenbau (hohe Produktkomplexität, hohe Variantenvielfalt, hoher Entwicklungsaufwand
- Konkrete Aussage zu den Arbeitspakten, die zum Erreichen der aufgezeigten Ziele und Teilziele erforderlich sind (Roadmap) inkl. Aufwandsschätzung / benötigte Kompetenzen.
- Darstellung der Vorgehensweise bei einer möglichen Ablösung der aktuellen Software

## 3.3. Sonstige Projektanforderungen

- Mind. 1 Berater/in, der die Analyse und das Vorgehenskonzept erarbeitet, sollte auch die Umsetzung in einem Folgeprojekt begleiten, damit wir uns vom Erstkontakt an auch ein Bild über die handelnden Personen machen können.
- Erfahrung im Mittelstand, Maschinenbau, Sondermaschinenbau (hohe Komplexität, geringe Stückzahlen, hohe Variantenanzahl)

ICL