# 2 Technical Proposal

## 2.1 Product Requirements

### 2.1.1 Technical Specifications

*The Application* will conform to the form, fit, and function of the {MaxPlez} v3.20 Beta release provided in the RFP package. It will also incorporate certain extensions and enhancements as specified within the RFP.

### 2.1.2 Features

*The Application* will maintain the current appearance and behavior of {MaxPlez} v3.20 Beta. It will include experimental setup, run, and analysis functionality.

{XC} commits to maintaining the following performance goals throughout during the development of *The Application*:

- Functional results will match {MaxPlez} v3.20 Beta. This includes algorithm performance and results, which must meet or exceed the performance level of {MaxPlez} v3.20 Beta.
- Visual presentation will follow good application flow and rules of progressive disclosure as exhibited by {MaxPlez} v3.20 Beta.
- A high level of control customization will be implemented, enhancing convenience and appearance.
- *The Application* will perform well in both common and uncommon user scenarios.
- *The Application* will not contain known application errors or bugs that impact final displayed results. This includes, but is not limited to, text and graphical errors relating to quantities, Ct's or final calls or other results as shown in the analysis result screens (e.g. Amplification Plots, Plate Sample Values, Text Report, etc.)
- *The Application* will not crash.
- All known application errors or bugs will be fixed prior to the release of *The Application* except for those deemed to be acceptable by {Strategic}. {XC} acknowledges that this number is typically extremely small (i.e. less than 10 for any newly found issues prior to release) with any remaining errors having minimal impact to usability.

### 2.1.3 Extensions

{XC} will implement all the extension features defined and documented in “Extensions to {MaxPlez} 3.20 Beta” in *the Application*.

### 2.1.4 Performance

*The Application* will meet or exceed the actual performance of the {MaxPlez} v3.20 Beta release software for both file-based experiment files and database-based experiment files.

## 2.2 Executive Summary of the Development Process

Unlike the development of some software products that has continually changed specifications, *The Application* has well-defined scope and specifications. It will maintain the same form, fit, function and features of existing {MaxPlez} v3.20 Beta release, with additions of certain extensions. To meet this requirement, {XC} will use a *conventional* software development process. The process includes following steps, with limited iteration added to the last period of the development cycle after alpha release.

1. Requirements analysis
2. Specification
3. Design and Architecture
4. Implementation and testing
5. Documentation (an on-going task)
6. Alpha and beta releases, (limited) interation between steps 1 to 6
7. Acceptance test, final release
8. Maintenance

### 2.2.1 Requirements Analysis and Specifications

This stage will define what will be done.

We will extract the requirements of *The Application* from existing {MaxPlez} v3.20Beta software, the "Extensions to {MaxPlez} 3.20 Beta" document, the existing User's Manual, and any additional documents provided by {Strategic}. Then we will develop the detailed functional description, use cases and specification for *The Application*.

In addition to reviewing the software and various documents, we will conduct intensive training for our group members about real-time quantitive PCR application. Building upon our close relationship with Zhejiang University, we will invite professors from The Institute of Biomedical Engineering to provide the training. Information about this Institute is available at http://www.cbeis.zju.edu.cn/bme1/Intro/INSTITUTE OF BIOMEDICAL ENGINEERING.htm.

During the software review, we will analyze, not only the {MaxPlez} software, but also the similar products offered by three to five of Stratgene's direct competitors. We have found that the competitive analysis is the best way to educate engineers; it immediately bring up a contrast comparison for the software feature and functions.

The {XC} team knows that for us, Requirement Analysis and Specification is a most critical task. While {XC} is very familiar with general instrumentation software design and implementation, we are less experienced in biology, genomics, proteomics and drug research. Once we thoroughly understand product requirements, we will carry out the project efficiently.

### 2.2.2 Design and Architecture

The design phase will include:

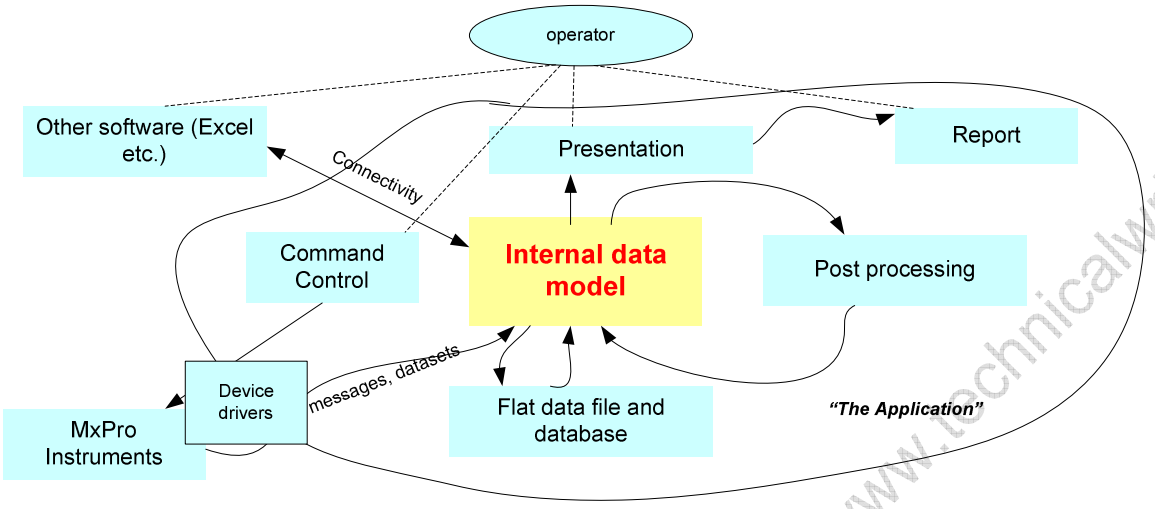
- Preliminary design and review
- Detail design and review
- System level design, architecture design
- List of sub-system design
- Unit design
- Algorithm design
- Communication design (message, control logic for the instruments)
- Major user interface design (requires artistic work).
- Testing plan for system, sub-systems and units

On the sub-system and unit level, the design phase will define the major property components and the interface functions (the *Methods*).

The following key points will guide the architecture design:

1. Make the entire system data centered. While people only see the user interface, the internal data model is the most critical component of the software. We will carefully and thoroughly design the data object property and the methods.
2. Clearly separate the user interface from the internal data model. This architecture has several advantages:
   - More easily introduce new user interface, such as web-based
   - Better multiple language support
   - Better multiple regional support (such as different engineering unit support)
3. Clearly separate the communication methods from the internal data model. Communication protocols must be independent of the data and commands being transferred.
4. Use state-machine and other methods to design the control logic so the hardware can be controlled with well-defined logic. The hardware should never run into dead-loop.
5. Take the advantage of *multi-thread, delegate* and *events* and other new Windows real-time technology in .NET to provide "flashing" and reliable instrument operation and data presentation
6. Maintain objected-oriented design and programming, modularity, software reusability and maintainability as important engineering criteria

A "data-centered" architecture will be constructed for *The Application*, as shown in the following illustration:




Contrary to some software engineers who believe the *Design* can be separated from the *Implementation*, at {XC} we maintain that in the design stage, it is important to determine the implementation technology to be used. For example, there are many ways to access SQL database server. During the design stage, defining the specific programming technology to be used in the coding makes for a much smoother implementation phase.

Furthermore, some user interface designs are essentially the prototypes of the implementation. Designs can be directly made within Visual Studio environment. Considering implementation within the design phase makes for an easier overall project.

As mentioned earlier, the design for *The Application* will include a companion testing plan. The automated testing method using Visual Test will be designed in parallel with *The Application* itself.

### 2.2.3 Implementation and Testing

The implementation includes the coding of *The Application* and the testing methods based on the design documents. Roughly 30% of the resources will be allocated to various testing.

Implementation will be conducted at unit, sub-system and system levels, in parallel. The initial phase will focus more efforts on the unit and sub-systems, while the ending phase will be more focused on the final integration at system level. Some modules and functions, such as reporting or file import and export, can be tested at sub-system level. Many other functions such as communication control can only be tested at the system level.

Peer reviews are internally conducted reviews focused on identifying defects in software development artifacts. We will include peer reviews to ensure that the code implements the allocated requirements and complies with project standards.

### 2.2.4 Documentation

Over the course of design, implementation and testing, various documents will be developed per ISO standard and the Regulatory Requirements defined in the section 7.6 of RFP. Previously, we have used

SourceSafe and SOS to manage engineering documents. SourceSafe has the advantage of revision control, check in/out function and Internet access capability. However, we intend to do a product comparison analysis to identify a better document management tool.

The software source code will have intensive in-line comments. We expect that 20 to 25% of the text in the source code will be in-line comments.

### *2.2.5 Alpha and Beta Releases, Final Acceptance*

**Alpha release**: After most of critical functions and modules (unit and sub-systems) are developed and tested, an initial internal release can be designated as the alpha version. Alpha version will be the initial version that can be reviewed internally in both {XC} and the {Strategic}. Alpha release is scheduled 6 months before the final acceptance. It will trigger a sequence of marketing activities, such as developing the website, marketing collaterals, brochures, demo software, Windows Help, User's Manual, etc. {XC} will actively cooperate with {Strategic} for these marketing development activities.

NCRs (Non Conformance Report) will be issued by the internal and external reviewers. {XC} will fix the required NCRs, using the standard development process outlined in ISO or CMM standard. Certain changes will be made to the original requirement, specifications and design documents. All documents will be maintained up-to-date. Coding will be conducted based on updated design requirement.

**Beta release** is expected 3 months before the final acceptance of *The Application*. Beta release is targeting at a wide range of marketing people and a group of Beta customer testing sites. Again NCRs will be issued by the reviewers.

**Final Acceptance**: One month before the target final acceptance date, a "final version" will be delivered to {Strategic} for final review and comments.

## 2.3 Preliminary Development Plan and Timelines

The development plan is established by creating timelines for each stage of the process. The following preliminary development plan shows the major activities and deliverables within the 18 month period. Payment will be checked against the status of the following timelines:

| Timeline | Main Activities | Deliverables and formats |
|---|---|---|
| Month 1 – Month 3 | ▪ Setup development environment: Computers and network, Visual Studio, SourceSafe/SOS, TestTrack, Visual Test, documentation management environment<br>▪ Start recruitment for all required personnel<br>▪ Intensive training about QPCR application<br>▪ Develop the requirement analysis, Define Software Specification<br>▪ Finish detail human resource allocation plan for the whole project<br>▪ Finish the preliminary design<br>▪ Initiate the architecture design | ▪ Software development environment description (Word)<br>▪ Recruiting status report (Word)<br>▪ Training status report (Word)<br>▪ Resource use plan (MS Project)<br>▪ Preliminary design (Word) |
| Month 4 - Month 6 | ▪ Finish the architecture design<br>▪ Finish the requirement analysis and specification definition<br>▪ Sub-system design<br>▪ Testing plan design<br>▪ Initiate subsystem implementation (coding)<br>▪ UI layout design | ▪ Architecture Design (Word)<br>▪ Sub-System Interface Design (Word)<br>▪ Requirement analysis and specifications (Word)<br>▪ Testing Plan (Word)<br>▪ Some programs for sub-system or units (Source)<br>▪ UI layout prototypes (Source, EXE) |
| Month 7 - Month 9 | ▪ Sub-system implementation<br>▪ Sub-system testing<br>▪ Initiate system integration<br>▪ Initiate system testing automation | ▪ Sub-system design documents (Word)<br>▪ Partial implementation for sub-system and general layout (Source, EXE)<br>▪ System integration status (Word) |
| Month 10 - Month 11 | ▪ Continue the sub-system implementation<br>▪ Continue the system integration<br>▪ Continue testing, verification and validation | ▪ Partial implementation for sub-system and system (Source, EXE)<br>▪ System integration status (Word)<br>▪ Testing and verification results (Word, VT) |

| Timeline | Main Activities | Deliverables and formats |
|---|---|---|
| Month 12 | ▪ Alpha release and review<br>▪ Cooperate the marketing release activities | ▪ Alpha software (Source, EXE)<br>▪ Alpha software description (Word)<br>▪ Recommendation for marketing releases (Word) |
| Month 13 - Month 14 | ▪ Change requirement and specification based on alpha version review<br>▪ Change the design and implementation<br>▪ Continue the sub-system and system testing | ▪ Updated software requirement and specifications and changing history (Word)<br>▪ Updated design documents and changing history (Word)<br>▪ Testing, verification and validation reports (Word, VT)<br>▪ Updated software buiSLD (Source, EXE) |
| Month 15 | ▪ Beta release<br>▪ Beta review | ▪ Beta software (Source, EXE)<br>▪ Beta software description (Word) |
| Month 16 - Month 17 | ▪ Review and testing the beta version; only minor changes to the requirement and specs are allowed<br>▪ Additional system testing<br>▪ Deliver complete CDRL documents | ▪ Beta software review description and action plan (Word)<br>▪ Testing status (Word, VT)<br>▪ CDRL documents (Word) |
| Month 18 | ▪ Deliver the software for final review<br>▪ Improve the CDRL per feedback<br>▪ Final acceptance review<br>▪ Project concluded | ▪ Final version software (Source, EXE)<br>▪ Updated CDRLs |

***Legend:***

**Word**: Text files such as Microsoft WORD, PDF, Excel, or PowerPoint
**Source**: Source code
**EXE**: executables
**VT**: Visual Test results

## 2.4 Contract Data Requirements List (CDRL)

At minimum {XC} will provide the following documents as Contract Data Requirements List (CDRL.) The description of each document is provided at the website: http://sparc.airtime.co.uk/users/wysywig/didlist.htm. CDRL documents are included in the deliverables.

The table below provides the preliminary initial release dates for these documents. Because they will be updated over the development period, the final versions will be delivered to {Strategic} together with the source and executable software for final acceptance. This will occur about 1 month before the whole project concludes. Initial released documents will also require the approval from {Strategic}.

| Item Number | Data Item Description ID | Title | Document Initial Release Date |
|---|---|---|---|
| A01 | DI-IPSC-81427 | Software Development Plan (SDP) | Month 3 |
| A02 | DI-IPSC-81433 | Software Requirements Specification (SRS) | Month 3 |
| A03 | DI-IPSC-81435 | Software Design Description (SDD) | Month 3 |
| A04 | DI-IPSC-81428 | Software Installation Plan (SIP) | Month 6 |
| A05 | DI-IPSC-81430 | Operational Concept Description (OCD) | Month 6 |
| A06 | DI-IPSC-81431 | System/Subsystem Specification (SSS) | Month 6 |
| A07 | DI-IPSC-81434 | Interface Requirements Specification (IRS) | Month 6 |
| A08 | DI-IPSC-81432 | System/Subsystem Design Description (SSDD) | Month 6 |
| A09 | DI-IPSC-81436 | Interface Design Description (IDD) | Month 6 |
| A10 | DI-IPSC-81437 | Database and File Format Design Description (DBDD) | Month 6 |
| A11 | DI-IPSC-81438 | Software Test Plan (STP) | Month 9 |
| A12 | DI-IPSC-81439 | Software Test Description (STD) | Month 9 |
| A13 | DI-IPSC-81440 | Software Test Report (STR) | Month 12 |

### *2.4.1 Approval Process*

Except for the alpha and beta releases, {Strategic} will review the delivered documents including various reports, CDRLs, source code and executables for completeness and provide feedback to {XC} electronically within 5 working days. {XC} will review {Strategic}'s feedback and incorporate changes within 2 weeks and provide updated copies electronically to {Strategic}. Receipt by {Strategic} of the revised documents shall constitute acceptance by {Strategic} and completion of the corresponding milestone.

For alpha and beta software releases, the time of {Strategic}'s review can be as long as 4 weeks. {XC} will review the {Strategic}'s feedback and incorporate changes within the next 4 weeks. NCRs (Non Conformance Reports) will be issued for the major bugs or problems identified in the alpha and beta release reviews.

## 2.5 Development Organization and Resource Allocation

The engineering team for *The Application* development will involve a total of nine people. Some of them will partially allocate their time to the project. Among them, the project lead, John Zhow, and Brill Qunnt,

the senior Windows consultant, are located in the Silicon Valley, USA. Others are located in Hangzhou, China.

The job function of each person for *The Application* project is listed below:

| | Name | Availability | Location | Time Usage | Main Job Function |
|---|---|---|---|---|---|
| 1 | John Zhow | Available | Surrey, USA | 25% - 75%, various | Project Lead, {XC} liaison. Defines the requirements and specs, manages the overall resources and schedule. Coordinates major activities |
| 2 | Brill Qunnt | Available | Surrey, USA | 25% - 50% various | {XC} long-term contractor, senior software consultant. Architecture and system design |
| 3 | Han Hong | Available | Hangzhou, China | 100% | {XC} China Team Project Manager for *The Application*<br>Design and implementation, manage the group in China |
| 4 | Jason Chen | Available | Hangzhou, China | 50% | System and sub-system design, some implementation work |
| 5 | (Senior Windows developer) | To be hired | Hangzhou, China | 100% | Sub-system design, implementation, testing |
| 6 | (Senior Windows developer) | To be hired | Hangzhou, China | 100% | Sub-system design, implementation, testing |
| 7 | (Marketing/QA Manager) | To be hired | Hangzhou, China | 100% | Marketing and QA. Must have QPCR application background. Help to define the requirement and specs, provide QPCR application education and guidance. Coordinate QA activities. Approve the functions, algorithms of implementation |
| 8 | (Testing Engineer) | To be hired | Hangzhou, China | 100% | Conduct whitebox and blackbox testing, implement automated test via Visual Test. |
| 9 | ISO/CMM Manager | To be hired. Recruiting started in Jan. 2006 | Hangzhou, China | 50% | Documentation, lead ISO/CMM certificate process at Radiant.<br>The cost of this person will not be charged to this project. |

The first four people are available immediately. (Their resumes are attached in the Appendix 1.) Brill Qunnt was the chief architect and project manager for the SLD .NET instrument software project. A few months ago he left SLD to become senior software consultant for {XC}.

Jason Chen and Han Hong have been working in the SLD .NET instrument software for years in China. We will move these resources to *The Application*.

We plan to hire two senior-level Windows developers with at least 3 to 5 years of working experience. Their main jobs will be sub-system design, coding and unit-level testing. If we were unable to find the qualified engineers within the time limit, we can shift some engineers from another group.

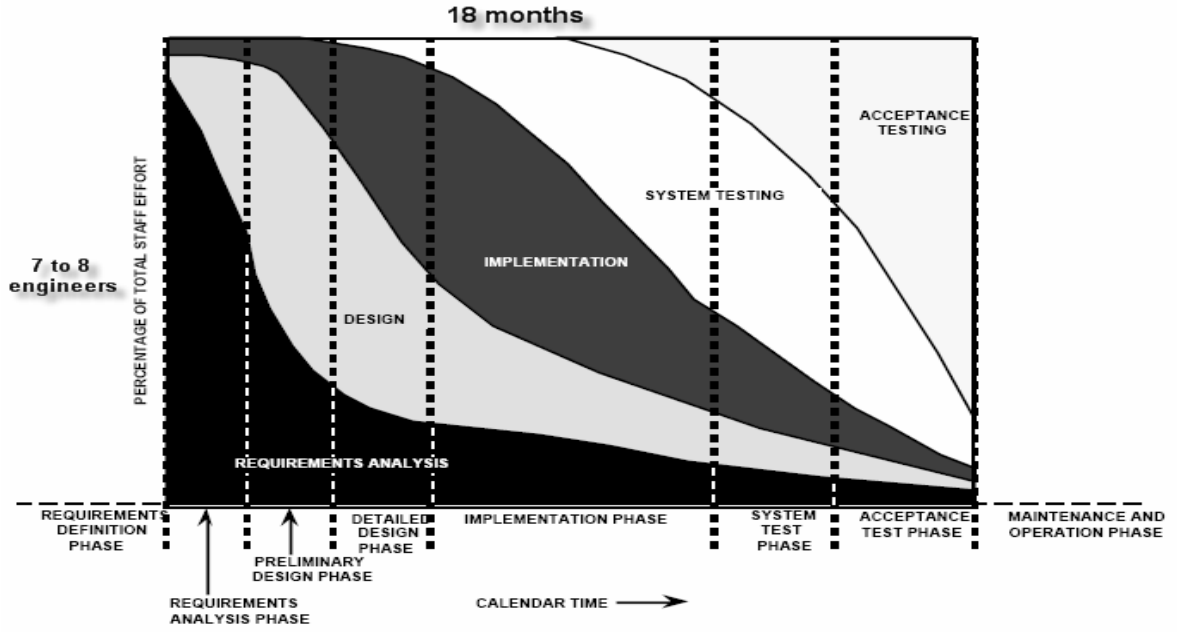
In the initial phase of design and implementation, Jason Chen, Han Hong and two senior Windows developers will each be focused on different sub-system developments. For example, one engineer will focus mainly on the user interface, while the other will focus on data modeling and file format. When the implementation nears the end, all engineers will work on both system-level and their own sub-system level aspects of the overall projects.

The Marketing/QA Manager will play a key role in the project. As discussed in the Strength, Weakness and Risk section on page 24, the {XC} team's expertise in the QPCR applications needs some additional help. We plan to fill this position with a person who has strong background in QPCR usage. This person must have a Ph.D. or Masters degree in biology or a related field. This person's role will be to help the engineers in China to understand the application, algorithm and various use cases. They will also coordinate the software quality assurance activities.

The testing engineer to be hired will also be required to have certain programming skills so that this individual can not only conduct the test but also program the Visual Test automated testing suites.

{XC}/China initiated the recruiting for an ISO/CMM manager in January, 2006. This person will dedicate substantial amount of time to *The Application* project. However, this individual's time will not be charged to {Strategic} because their work is not application-specific.

The human resource allocation of the project from start to first release is shown in the following figure:




This diagram illustrates that there will be no "absolute" boundaries between each stage of the process. The main role of the Project Lead is to optimize the time and human resources to gain the maximum benefits throughout the project.

## 2.6 On-Going Maintenance Plan

After the initial release for *The Application*, on-going sustaining engineering will be provided. {XC} will apply efforts to:

1. Fixing urgent bugs and generating patches; we do not expect this will happen frequently but it does happen
2. Continuous improvement of and enhancement to the software and generating new major releases
3. Creating special individualized versions for customized use

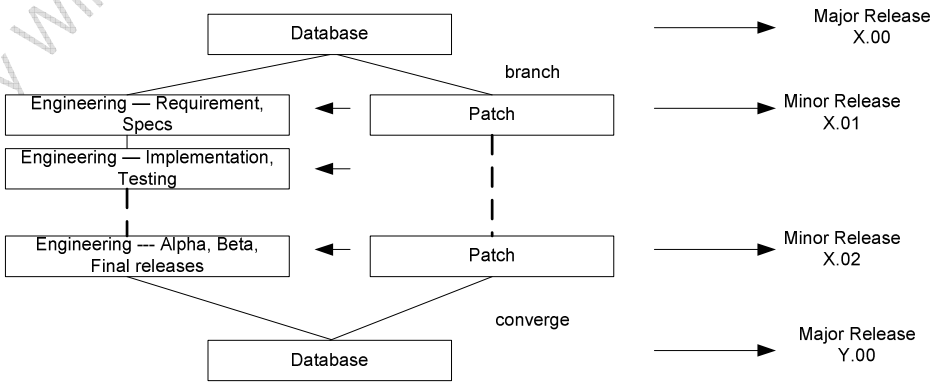
Release Management will be based on four main functional blocks

- Change Management
- Build Management
- Deployment
- Incident Management

The process outlined below implements release management by iteratively increasing the capabilities and interactions of these four blocks.


Version control provides the means to track the actual source changes so that any issues raised after a release can be traced and fixed. Version control is the core of release management.

{XC} plans to adopt its successful version branching technique during the sustaining period. After each major release (which happens only once or twice a year), the software source database is branched into two databases: *engineering database* and *release database*. If there is an urgent bug that has to be fixed and the field software must be updated, the bug will be fixed in both databases while a patched version will be sent out from the *release database* after QA. This case is very rare but it does happen. In parallel, the engineering database can continue through the regular software development process to meet the goal of the next major release. Before the next major release, two databases will be cross-checked and converged. This process can be shown in the chart below:



We recommend that *The Application* adopts a fixed time schedule between major releases. Unlike the Internet industry which is more competitive and dynamic and where the release period can be as short as a few months, the instrumentation industry can probably accept annual release. This approach is subject to further discussion with {Strategic}.

With this assumption, we plan to have the major release ready by the end of each year. The scheduled major release dates are as follows:

| Aug 2007 | Nov 2007 | Aug 2008 | Nov 2008 | Aug 2009 | Nov 2009 |
|---|---|---|---|---|---|
| Beta release | Final Major Release | Beta release | Final Major Release | Beta release | Final Major Release |

Minor releases will be made between major releases to provide urgent patches and customized versions.

## 2.7 Development Environment

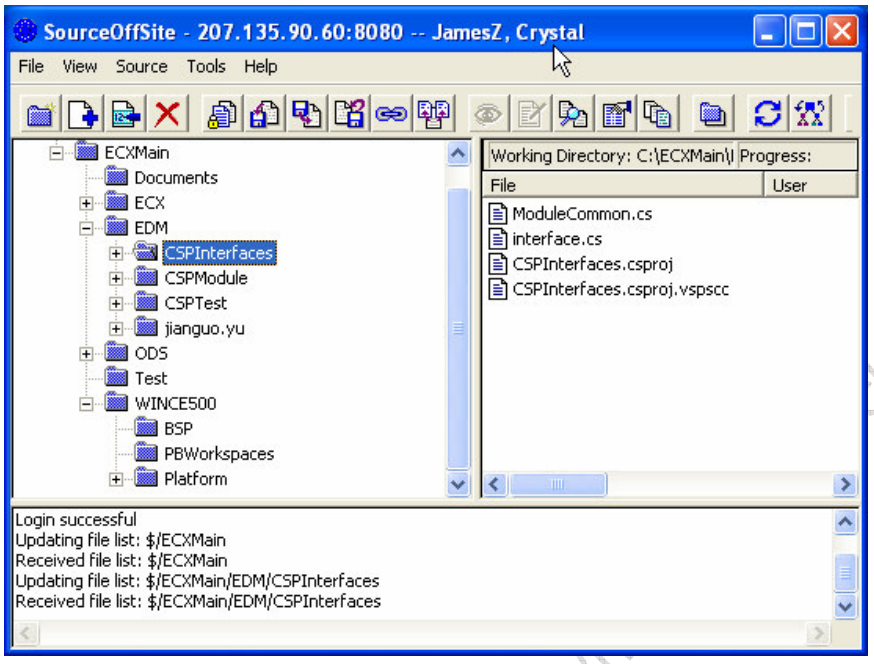
### *2.7.1 Programming Environment*

We will use Microsoft Visual Studio 2005 with .NET framework combination of C#, XML, VB.NET, ADO.NET and/or MFC. C# will be the main programming language while the others will be used to provide flexibility and performance improvement.

### *2.7.2 Version Control*

In the past, {XC} has adopted different version build strategies for different project. For *The Application* development, we plan to adopt a daily build or weekly build strategy during the development stage. This means a current version will be built daily, but no longer than weekly, for internal review and testing.

We will use the MS-Visual SourceSafe tool for version control and for the team collaboration platform. It is completely integrated with the .NET programming environment and comes with MSDN subscription. For remote operation, we will use SourceSafe Offsite (SOS) from SourceGear (http://www.sourcegear.com/sos/). The SOS is an Internet-based tool that allows users to access the Visual SourceSafe database remotely. We have been using this Internet tools for many years. A screen copy of SOS that we currently use is attached below:



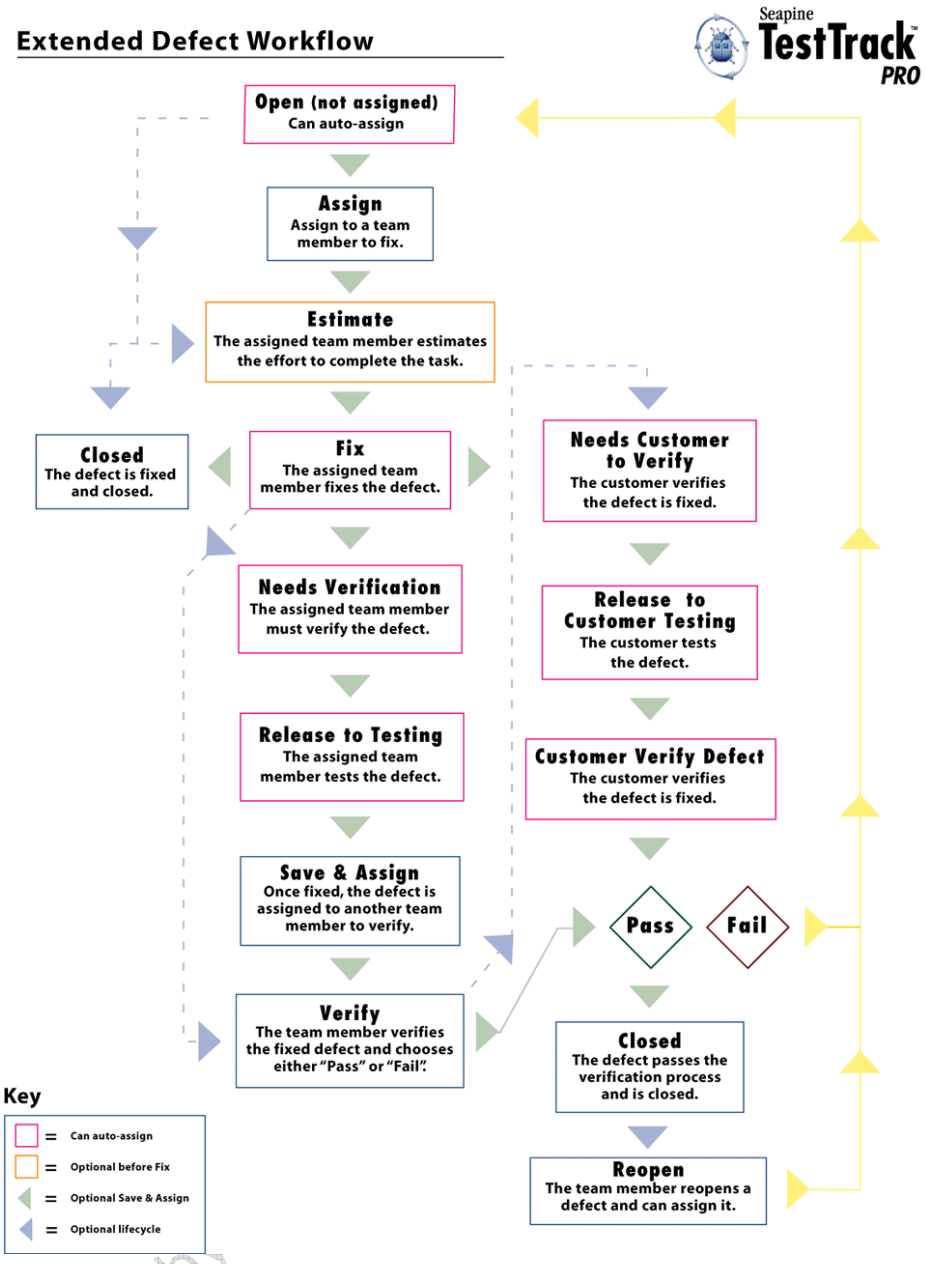

Note that the building strategy for software maintenance is different from that in the development stage. It will be explained in more detail in the Maintenance plan.

### 2.7.3 Bug and Enhancement Feature Track

For *The Application* development and maintenance, we will use the TestTrack Pro from SeaPine (http://www.seapine.com/ttpro.html) as our bug-tracking management software tool. {XC} team has used this tool extensively in the past, and we like its capability.

TestTrack will become the central task coordination system for the project lead, developers and testing engineers in the group. The project lead will play a key role in coordinating the tasks using this bug tracking system. On a daily basis, he or she will inspect the progress of each individual with assigned task list and make de{XC}sions about each item.

The workflow of using TestPro which {XC} team has strictly adopted in the past years for the other projects is described in the following chart:




In addition to TestTrack Pro, we will use Microsoft Project for overall project and resource management. We have found that Microsoft Project is a good complementary tool to TestTrack Pro in project management. Microsoft Project will track the tasks on larger scale, while TestTrack Pro will address the individual tasks and details.

### 2.7.4 Automated Software Testing

{XC} engineers have used Visual Test from Rational (acquired by IBM) to conduct automated software testing. We found that Visual Test is a powerful complementary tool for regular manual testing. Visual Test can record the computer actions of the operator and play them back automatically and repeatedly.

Certain criteria can be preset by the Visual Test programmer and used to check against expected value or messages.

We plan to use Visual Test for testing automation for *The Application* in addition to the required manual test. As we have found previously, during the software development cycle, more and more manual testing cases will be integrated into the Visual Test script. Manual test will focus on newly added features during the development. We have found that the automated test tool can catch more than 80% of unexpected bugs. It is particularly useful when daily or weekly build strategy is adopted.

### 2.7.5 Deployment and Installation

*The Application* will support {Strategic}'s Mx3000P, Mx3005P, and next generation instrument platforms. It will be deployable on the Windows XP, Windows 2000 **and Windows Vista** operating systems, running on Microsoft-supported Intel-based PC laptop and desktop computers. *The Application* will be compatible with Microsoft SQL Server 2005 and Microsoft SQL Server Express 2005. *The Application* will be deployable in both flat-file and database versions from the same CD-ROM.

The installation will be managed by the InstallShield.

## 2.8 Regulatory Requirements

### 2.8.1 Regulatory Requirements

All project development work will conform to the FDA Quality System Regulation, 21 CFR Part 820[1], and to Directive 98/79/EC of the European Parliament and of the Coun{XC}l of 27 October 1998 on in vitro diagnostic medical devices[2] (In Vitro Diagnostics Directive).

A comprehensive design history file will be developed and maintained for *The Application* per requirement defined in the section 7.6 of RFP. The design history file will be maintained for the life of *The Application* and include additional documentation for each subsequent modification or addition in accordance with 21 CFR Part 820.

### 2.8.2 Coding Standards

When applicable, the coding standard will conform with the convention defined in the document attached to the RFP, "DRAFT GUI Software Coding Standards.". When not applicable, {XC} will use the coding standard recommended by Microsoft: http://msdn.microsoft.com/library/?url=/library/en-us/vsent7/html/vxconCodingStandardsCodeReviews.asp.

## 2.9 Resource and Support Required from {Strategic}

{XC} will require the following resource and support from {Strategic}:

- Resources that review and approve each deliverables per Approval Process

- Loaner hardware equipment for development, two sets for each type of instruments, one set to be located in Hangzhou, China and one set in the Silicon Valley
- At least six beta site customers

## 2.10 Strength, Weakness and Risk

### *2.10.1 Strength*

For this project we believe that the following items are the {XC} team's strongest points:

#### *Strong Background in General Instrumentation and Measurement System*

We have been working in the instrumentation industry for many years. We have developed and launched a number of successful software products into the market. We understand the development cycle and have hands-on management experience in the complete product development. We understand not only the Windows development, but also device driver, firmware, microprocessor, data acquisition signal processing and algorithms. While {MaxPlez} system is designed specifically for QPCR usage, from the technical standpoint it can be divided into various sub-elements that are common to those we have completed for other instrument software.

#### *Suitable Skill-set in Windows Programming*

Our largest client SLD has gone through a similar software re-write process and {XC} was contracted with the project. Thus, we already have the required experience in software porting. The {XC} team is experienced in both MFC and .NET environment. We have a thorough understanding to how to apply the technology which was previously implemented in the application in MFC or plain C into the newer technology in .NET.

Having reviewed the {MaxPlez} software, our engineers have concluded that *The Application* is probably 1/3 of the scale of our SLD project, and it is technically less difficult. *The Application* does not ask for high speed data throughput, hundreds of fancy display types and sophisticated real-time control logic as did those we developed for the SLD. The graphics, reporting, network connectivity, multiple thread operation, signal processing are relatively easier to implement. In Appendix 2, {XC} Capability Demonstration on page **Error! Bookmark not defined.**, we compare a few dozen functions that are part of our experience with those in these individual areas. These experiences will be most helpful to this project.

#### *Small but Focused*

Though the {XC} team has provided engineering services to many clients in the past, currently it has only one main client, SLD. If {XC} wins this contract, {Strategic} will become the second largest customer of {XC}. Given the significance of this stream of revenue, {XC} will give complete attention to this project and take its success most seriously. {XC} intends to build a long term relationship with {Strategic} from this project.

#### *Offshore Resource with Local Management*

{XC} has offices in both Sunnyvale, California and Hangzhou, China. While the main development resource is in China, the interface and management is local to {Strategic}. Our offshore development provides cost advantage, and our management in US maintains fluent and effective communication. This structure provides better assurance to {Strategic} about both product quality and intellectual property protection.

### *2.10.2 Weaknesses*

We acknowledge that the following items are the weak points of {XC} team:

#### *Lack of QPCR Application Background*

The {XC} team is new to the life science industry. Before we initiated conversations with {Strategic}, we knew little of QPCR. We are the application background of molecular biology, genomics, proteomics, drug discovery where QPCR instruments are used. We fully agree that the engineers must have an application background, in order to design and implement a good product. In order to overcome this weakness, we will take two actions:

1. Conduct intensive training to everybody in the group about basic theory of biology and how the QPCR technology is applied. This will take place in the very first phase of the project
2. In China **we will hire a person with academic degrees in biology** or a related major. This person will provide guidance to the software developers and also act as a quality assurance engineer.

#### *{XC} is not ISO or CMM Certified*

{XC} is not ISO 9001, ISO 13485 or CMM certified. We have been talking about the possibility of becoming certified for years. However, since the demand has not been urgent, we have not pursued certification aggressively.

Although {XC} was not ISO or CMM certified, {XC} follows the essential principles of the software development process. This includes various documentation and required processes. For example, we demand all the projects or sub-projects must have design documents before implementation is started. We have strictly followed the bug-tracking work flow to report, monitor, implement, verify and approve each individual task.

If {XC} wins this contract, we will accelerate the certification process. **{XC} is committed to pass ISO or CMM-3 before the beta release of *The Application*.**

In January 2006 the {XC} China engineering team has initiated recruitment for a certified quality assurance manager who will eventually lead the company to pass the ISO or CMM certification. This activity is independent of this project but definitely will be helpful to it.

### *2.10.3 Risks*

We do not see significant risk for this project because:

1. The scope, requirement and the specification of the project are well defined. The behavior and the performance of the hardware are clearly known. There is little uncertainty in the definition.
2. The programming technologies are readily available, and the {XC} team is well qualified to use them.
3. Payment is against the combination of deliverables and reached milestones. We anticipate no big surprises.

Furthermore, to reduce the risk to {Strategic}, {XC} has scheduled the alpha software release at about 6 months before the final acceptance deadline. This schedule gives sufficient time for disaster recovery if unexpected events occur.