



The Value of Migration

A White Paper on dealing with the end of support for
proprietary SilverStream Classic

Infoioia SA, 2005



Executive Summary

SilverStream/Novell has ceased all support for running applications built with SilverStream-proprietary technologies.

Users of SilverStream-proprietary technologies thus face the urgent question as to how to sustain the evolution and smooth running of their SilverStream-proprietary applications.

This White Paper discusses the advantages and disadvantages of four alternative scenarios:

- Maintain the SilverStream application(s) "as is"
- Redevelop the application(s) in other proprietary technologies
- Redevelop the application(s) completely in the J2EE standard (Java 2 Enterprise Edition)
- Migrate from SilverStream to the J2EE standard

Assessment Criteria

In the interest of internal/external clients, the key criterion in assessing alternatives is to make sure a stable and fully functional application is guaranteed to go live as soon as possible.

The other criteria used in this White Paper are:

- Evolution (scalability, technology integration...)
- Costs
- Progress of current application(s)
- Compatibility with other technologies
- Degree of dependence on proprietary technology
- ROI in existing know-how and functional business logic
- Training requirements

End of Support for SilverStream Technologies

The major strength of the SilverStream Application Server versus products from larger suppliers such as IBM and BEA has been an enormous productivity advantage in developing internet-based applications. SilverStream clients have profited by having their applications operational faster, cheaper, and more feature-rich.

The downside is the dependence on a proprietary technology. In fact, when the product first came to market, there were no standards for this type of application. In the past years, however, the area of "internet-based application development with Java" has been standardised and grouped under the header and abbreviation J2EE.

Already prior to its takeover by Novell in 2002, SilverStream had announced that it would support applications built with proprietary SilverStream technologies only for a limited period of time.

As a result, SilverStream customers investigate alternatives available to sustain their business applications built with SilverStream-proprietary technologies.

Alternatives

Alternative 1: Maintenance of the SilverStream application(s) "as is" The alternative requiring the least amount of action is certainly to maintain the current SilverStream application(s) "as is". The SilverStream server is generally stable, and there is in-house experience with the existing product. Still, maintaining the *status quo* is a risky undertaking.

Experience shows that new errors, previously unknown, may appear in operation which can only be removed by the manufacturer. It is risky to rely on a static SilverStream server in a rapidly developing IT landscape.

Furthermore, it would not be possible to follow new developments in the area of the very dynamic J2EE standardisation, whereby one of the major advantages of the SilverStream Application Server platform would get lost, namely the possibility to integrate third-party applications.

Any further investment in the current project may be a step into a dead-end. Also, one depends on experts in an obsolete technology who may be increasingly difficult and costly to obtain.

Table 1: Advantages and disadvantages of alternative 1: Maintenance "as is"

PROS (+)	CONS (-)
<ul style="list-style-type: none"> • SilverStream server is generally stable • In-house experience with current product 	<ul style="list-style-type: none"> • New errors, previously unknown, may only be solved by manufacturer • Impossible to follow new J2EE developments and to integrate third-party applications • New project investments may be a step into a dead-end • Dependence on experts in obsolete technology

Alternative 2: Redevelopment in other proprietary technologies The second alternative entails the *complete abandonment* of the current application(s) and the development of (a) new application(s) using other proprietary technologies.

Typically, the time required to provide the same functionality and stability in an alternative application is of the same order of magnitude as has been invested in the current application. It is thus likely that over any significantly shorter period, a new system will either confront the users with feature limitations and/or stability issues.

To avoid user dissatisfaction in case of system failure, new applications should not be introduced until they have attained the feature-richness and stability of the current application(s).

Experience shows that it may take years to stabilise an application and make it perform well and stable under full or peak loads.

Table 2: Advantages and disadvantages of alternative 2: Migration to other proprietary technologies

PROS (+)	CONS (-)
<ul style="list-style-type: none"> • Depending on application size, more or less profitable 	<ul style="list-style-type: none"> • New application available by when? • Stability issues • Feature limitations • Stagnation of current application(s) at the expense of developing a new application • New training investments (time and money) • Loss of know-how and experience contained in the current application(s) • Full licensing costs • Possibly lesser integration of third-party applications • Again, dependance on proprietary technology, specialised staff and vendors

Factors that may impact on the waiting period for other proprietary solutions include:

- Volume of necessary analysis and implementation
- Available staff for the above analysis and implementation
- Knowledge and experience of staff
- Recruitment of experts in new technology and/or training of existing staff
- Tools used for development
- Available proof of stability and performance in production environments

For each new proprietary technology, a feature-by-feature comparison with the current application would have to be undertaken to ensure that all specifications and requirements are met. The time input for this assessment itself has to be taken into consideration when estimating the availability of the new application.

Still, even if estimates forecast availability of the new application within a reasonable timeframe, it entails a risk factor with potentially important implications (e.g. client dissatisfaction, loss or disruption of business, claims for damages, etc.) in case of non-availability.

Furthermore, it would be difficult to argue for any further development of the current application(s) while developing an alternative application to take over. The current application(s) would stagnate and client satisfaction and bonding may be put at risk.

Full licensing costs would have to be considered. Integration of such other proprietary technologies may prove to be difficult, and one again depends on proprietary technology, specialised staff and vendors.

**Alternative 3:
Redevelopment of
the application(s)
completely in the
J2EE standard**

The third alternative is the complete redevelopment of the application(s) using pure J2EE technologies.

J2EE is an industry standard, supported by a multitude of vendors and a large open-source community that provides virtually instant programming support. Already in 2003, about 50% of all application servers used online were either available under a free open-source license or at minimal cost (Source: <http://www.netcraft.com>, April 2003), providing excellent potential for cost savings.

Table 3: Advantages and disadvantages of alternative 3: Redevelopment in the J2EE standard

PROS (+)	CONS (-)
<ul style="list-style-type: none"> • Non-proprietary, evolutionary technology • No license fees • High compatibility with other technologies & applications • No dependance on external specialised support • ROI on existing Java know-how 	<ul style="list-style-type: none"> • Prohibitive time requirement • Stagnation of project progress

On the other hand, assuming a development effort for the original application(s) of several man-years, and considering some accelerating factors (we now know better what we want) and slowing-down factors (there are no J2EE development tools with which a productivity similar to SilverStream can be gained), one easily arrives at a prohibitive time requirement for this alternative.

While full resources are employed on the redevelopment, the project and features themselves would, however, stagnate.

**Alternative 4:
Migration of the
application(s) to
the J2EE standard**

The fourth alternative is to migrate the SilverStream-specific applications to J2EE-conforming Java applications.

Table 4: Advantages and disadvantages of alternative 4: Migration to the J2EE standard

PROS (+)	CONS (-)
<ul style="list-style-type: none"> • Much shorter time to market than alternatives 2 and 3 • Future-safe and easy-to-maintain application • Scalable and evolutionary architecture • Safeguarding of investments in proven business logic • Client satisfaction maintained • Current application(s) can continue to evolve • Cost reduction potential on license fees, or continuing support for the J2EE standard • Safeguarding of previous training investments • Step-by-step migration • New technology know-how • High compatibility • Reduced dependence on specialist labour and vendors 	<ul style="list-style-type: none"> • Not free (but less expensive than other alternatives)

This procedure has a number of advantages, which, combined with project-specific factors, make it very attractive:

There is a much shorter time to market versus alternatives 2 and 3.

Investments in several years of successful development of proven business logic are safeguarded.

Internal and/or external client satisfaction is maintained as there would be no decrease in service level. The current application(s) can continue to evolve without the risk of going into a dead-end. J2EE is an unlimited technology that is being propagated and developed by SUN and many other manufacturers.

J2EE offers maximum compatibility: the modular architecture of J2EE allows to integrate complete application modules into other applications and technologies.

A cost reduction is possible if the SilverStream server software is dropped (independence of a proprietary product); instead any J2EE server could be used (see www.theserverside.com/reviews/matrix.jsp for several dozen such alternatives).

Previous training investments are safeguarded since both SilverStream Classic and J2EE are based on Java.

Migration could be carried out step-by-step or partially to allow "soft" passage from the current to the future system.

New know-how about state-of-the-art architectures, technologies and standards can be obtained.

Dependence on staff and suppliers with very special know-how is reduced.

Conclusion

The transfer of current application(s) to the J2EE standard is likely to be the safest alternative.

Not only is it the safest alternative, it also takes advantage of the pros of the other alternatives while removing their disadvantages: the J2EE standard makes sure the new application is fully compatible with other technologies, is evolutionary and scalable, enhances return on existing know-how and experience, requires minimal staff training, and increases independence of third-party technologies and contractors.

Contact Information

For more information about Infoioia SA and migration software, technologies and training, please contact:

Wolfgang Gehner
Infoioia SA
Rue de Berne 7
CH-1201 Geneva
Switzerland

Tel: +41 22 900 00 09

Fax: +41 22 900 00 18

E-mail: wgehner@infoioia.com

www.infoioia.com