An Elementary Review of Linkages & Gaps Among BPR, SOA & Software Reverse Engineering

1,Prasenjit Kundu, 2,Bikram Keshari Ratha, 3,Debabrata Das

[1,]Research Scholar, Department of Computer Science, Utkal University
[2,]Faculty, Department of Computer Science, Utkal University
[3,]Lecturer, Bengal School of Technology & Management

Abstract:

Research on Business Process Reengineering has started as an effort in proposing methods and techniques for organizational restructuring in order to increase organizational efficiency and effectiveness. Later on researchers and consultants started using the tools of IT as part of BPR processes and ultimately they started to develop IT based models and methodologies for BPR. One such effort is the development of Object Oriented Knowledge Based model or the OKB model of BPR which encompasses the idea of Service Oriented Architecture (SOA) and Reverse Engineering in order to implement BPR in a simple yet efficient manner. In this paper an elementary literature review has been done in order to build a linkage among the concept of BPR, SOA and Reverse Engineering as well as to identify the gaps among these from the point of view of the OKB model.

Keywords: ADRI Approach, Business Process Reengineering, Dynamic Plug in instrumentation, OKB Model, Service Oriented Architecture (SOA), Software Reverse Engineering, Software Visualization.

1. Introduction

As the name of this paper suggests, this paper is basically an elementary review of literature because the review has been done from the point of view of the OKB model of BPR. Secondly the sources of this review are mostly the free resources available on the internet through different open access database and repositories like the Google Scholar, Social Science Research Network (SSRN), Open Access Research Database (OARD), Directory of Open Access Journals (DOAJ) etc. So this review is not a complete one but an elementary one.

2. Review on BPR

The emergence of global competitiveness in developing high quality low cost products has led to the development of various approaches like CIM, JIT, lean manufacturing, concurrent engineering etc and in this way the research on BPR and enterprise modeling has been started by researchers like J.H.Manley, K.C. Hoffman, H. Rozenfeld, A.F. Rentes, W. Konig, R. P. Anjard, S. Jarzabek, W. L. Tok, S. Kim, K. Jang, K. Mertins and R. Jochem([1], [2], [3], [4], [5], [6], [7], [8]). J.H. Manley was the first person to through light on reengineering manufacturing systems and connecting BPR with enterprise modeling. Manley's thought describes the ways through which industrial engineers can assist in reengineering worn out, error prone or obsolescent real-time manufacturing systems (embedded systems) by helping computer systems and also how the communication engineers can ensure that critical information control loops, both feed forward and feedback, are complete and efficient. Manley describes two conceptual models, the Embedded Computer System (ECS) physical model and the Object Transformation Process Model (OTPM) to guide a modified process flow analysis (PFA) of existing large-scale, complex embedded systems that takes into account the process-supporting information. As per Manley this modified PFA should be called an Information Process Flow Analysis (IPFA) [1]. Using an enterprise-wide Information Systems Architecture (ISA) as a roadmap, K.C. Hoffman presents a general model of the management structure to implement a systems integration program. This Information Systems Architecture (ISA) is supported by a defined set of measures and metrics and (ISA) approach is comprehensive in covering application software and data architecture and also the computing and communications hardware infrastructure and other automation technologies that support the overall business process [2]. On the basis of these two early approaches of enterprise modeling, J. H .Manley developed his marvelous idea of a three-phase information system analysis and design methodology which can be used to continuously improve enterprise information systems as part of a six-step annual business improvement process. According to this new approach, after the senior management's strategic decisions on next year's product and/or service portfolio content, the related financial, management, engineering, and quality improvement processes are analyzed to determine their output product and/or service quality and timeliness. Next, the facilities, equipment, and personnel resources required for individual processes are inspected for either immediate or possible future improvement. Next the minimum essential information (MEI) requirements are analyzed using the Object Transformation Process Model (OTPM). Next, the individual OTPM models are linked to help identify all pertinent data sources, information destinations, and timing requirements and lastly the linked OTPM models are mapped onto an Embedded Computer System (ECS) model that defines a physical architecture for improving telecommunication paths between all humans, machines and embedded

computers that are component parts of the integrated processes. Manley thinks that this approach yields comprehensive information system logical and physical architectural models that can recursively guide high-leverage enterprise-wide improvement projects over succeeding fiscal years [3].

3. Review on Software Reverse Engineering

Software reverse engineering is defined as the process of analysing a system to identify the system's components and their interrelationships, and to create representations of the system in another form or at a higher level of abstraction [9].Researchers believe that the interest in software reverse engineering has grown over the last twenty years due to the fact that comprehending and modifying software is at the heart of many software engineering tasks and this is also true in case of software reverse engineering.[10].An early example of the use and application of software reverse engineering is the famous case study by E. J. Byrne. Byrne conducted an experiment to reverse engineer a program and later documented the lessons learned from the experiment as a case study. Byrne used a reverse engineering process as part of a project to develop an Ada implementation of a FORTRAN program and to upgrade the existing documentation. To achieve this project objective, first the design information was extracted from the FORTRAN source code and entered into a software development environment. Then the extracted design information was used to implement a new version of the program which was written in Ada. Byrne reported the experience gained during this study and revealed issues about recovering design information, such as, separating design details from implementation details, dealing with incomplete or erroneous information, traceability of information between implementation and recovered design, and re-engineering [11].Erich Buss and John Henshaw mentioned Byrne's case study on reverse engineering and outlined their own initial research experiences on the application of reverse engineering technologies to large-scale legacy software systems and they realized that this application offers the opportunity to regain a measure of control and understanding of the software which can leverage the software developer's actions and knowledge, and augment the software development and maintenance processes[12]. G. Canfora, A. Cimitile & P.B. Lucarelli observe that it's difficult to define reverse engineering in rigorous terms because the field is new and rapidly evolving. They believe that traditionally reverse engineering has been viewed as a two step process and the steps are information extraction and abstraction .Information extraction deals with the analyses of subject system artifacts , primarily the source code to gather raw data whereas information abstraction deals with creation of user oriented documents and views[13]. Canfora, Cimitile & Lucarelli also support Chikofsky and Cross's definition of reverse engineering that is " the process of analyzing a subject system to identify the system's components and their interrelationships and to create representations of the system in another form or at a higher level of abstraction" ([13],[9]). Dynamic program analysis is the analysis of computer software that is performed by executing programs built from that software system on a real time basis. For dynamic program analysis to be effective, the target program must be executed with sufficient test inputs to extract and produce interesting behaviour. Dynamic program analysis helps to make a computational system reason automatically (or at least with little human assistance) about the behaviour of a program and draws conclusions that are useful to help the software developers to determine exploitability of vulnerabilities or to rapidly develop an exploit code [14]. Dynamic analysis produces output, or feeds into a subsequent analysis, that enables human understanding of the code and makes the design and testing task easy for the developers. Dynamic program analysis approach attempts to tune the application software during execution without stopping, recompiling or even rerunning the application. To achieve this objective it is necessary to use dynamic instrumentation techniques that allow the modification of the application code on the fly [15]. Program instrumentation is a general way to understand what an executing program is doing [16]. The principle of dynamic program instrumentation involves deferring program instrumentation until it is in execution and then inserts, alter and delete this instrumentation dynamically during the actual program execution. The Paradyn group at the University of Wisconsin and University of Maryland first used this approach to develop a special API that supports dynamic instrumentation and the result of their work was called DynInst API. DynInst is an API for runtime code patching that provides a C++ class library for machine independent program instrumentation during application execution. It allows attaching to an already running process or starting a new process, creating a new piece of code and finally inserting created code into the running process. The next time the instrumented program executes the modified block of code i.e. the new code is executed and the program being modified is able to continue its execution and does not require to be recompiled, re linked or restarted [16].

4. Review on SOA

Service oriented architecture can be defined as a framework to integrate business processes and supporting IT infrastructures into secure, standardized components services that can be reused and combined to address changing business activities and priorities[17]. For the last couple of years different organizations across different sectors are adopting SOA just because of its capability to ensure business process improvement and thus in this way to gain competitive advantage [18]. The reason behind strong adaptation of SOA is its link between IT and business and through SOA, services which require human interaction can be configured easily and systematically so that the whole system operates as per the operational needs. For this reason SOA reference architecture has become a clear framework for any enterprise operation [19]. According to Newcomer and Lomow, SOA should be seen as a style of design that provides guidance of creating and using business services

throughout their lifecycle as well as defines and makes provisions for the IT infrastructure that allows different applications to exchange data and participate in business processes seamlessly irrespective of the operating systems or programming languages underlying those applications. The elementary SOA ingredients are (1) Infrastructure, (2) Architecture (3) Process and (4) Governance [20]. According to Kraf-zig, Banke and Slama, SOA is based on four key abstractions: (1) Application Front Ends (2) Service (3) Service Repository and (4) Service Bus or Enterprise Service Bus (ESB).Actually the Application Frontend is the owner of the business process while the services provide business functionality which the application frontends and other services can use. A service consists of an implementation that contains business logic and data, a service contract and a service interface. The service contract specifies the functionality, usage and constraints for a client of the service and a service interface physically exposes the functionality. The service repository stores the service contracts of the individual services of an SOA and the service bus (ESB) provide the linkage between the application front ends and services. A client can either be an application front end or service. It is always an application front end that initiates a business process and receives the result. Service is a software component of unique functional meaning that typically encapsulates a high level business concept. The application frontend basically interacts with the human user and that's why it is called sometimes the client. In some cases the client can be the service itself. The client uses the interfaces to invoke the service through the ESB (enterprise Service Bus).The interface is based on the service contract and the service contract describes the service and the service fulfills the service contracts and the service contract is stored in service repository [21]. This framework can finally provide a basis for end user build front ends & applications developed according to Web 2.0 service definitions, such as mashups and the quick deployment of user oriented applications created in AJAX or similar development tool [19].

5. Review of Linkages Through The OKB Model Of BPR

Kundu, Ratha and Das's paper on Business Process Reengineering (BPR) is the cornerstone of this review paper. In their paper Kundu, Ratha and Das propose a mixed model of BPR by combining the OOM and KBM and call it the Object Oriented Knowledge Based Model or OKB Model [22]. The OKB model first identifies the business processes at the top or strategic level, middle or supervisory level and bottom or operational level and then breaks down these processes into repetitive activities. Then, the OKB model converts these repetitive activities into services as per the SOA framework. Once an enterprise wide SOA implementation blueprint gets ready, dynamic program instrumentation techniques are used to fine tune, optimize and reverse engineer the existing legacy systems across the organization [22]. Kundu , Ratha and Das's OKB model is unique in the sense that it uses the concept of SOA and dynamic program instrumentation, i.e.; it combines the principles of organizational reengineering with the tools of ICT and thus builds an effective, easy to understand and easy to implement framework for business process reengineering. The Object Oriented Knowledge Based (OKB) model has eleven steps and the steps are as follows :(Ref. Fig. 1)

[1] Departmentation of the Organizations.
[2] Identification of the process at three levels of management i.e. at the top or strategic level, at the middle or tactical level and at the bottom or operational level.
[3] Classifications of the processes into a) Core Process b) Associated Process and c) Auxiliary Process.
[4] Linking of processes or establishing the interdependency, Inter relationship and Inter connectedness among the processes.
[5] Conduct Information Process Flow Analysis (IPFA) using the ECS and OPTM
[6] Identify the services among the process. Step VII: Analyze the capability, usability of the existing legacy systems in the organization.
[7] Remove the systems that are totally obsolete.
[8] Design, Deploy New Systems using Service Oriented Architecture (SOA).
[9] Reengineer the partly obsolete systems using dynamic program instrumentation.
[10] Integrate the new systems with the reengineered one.

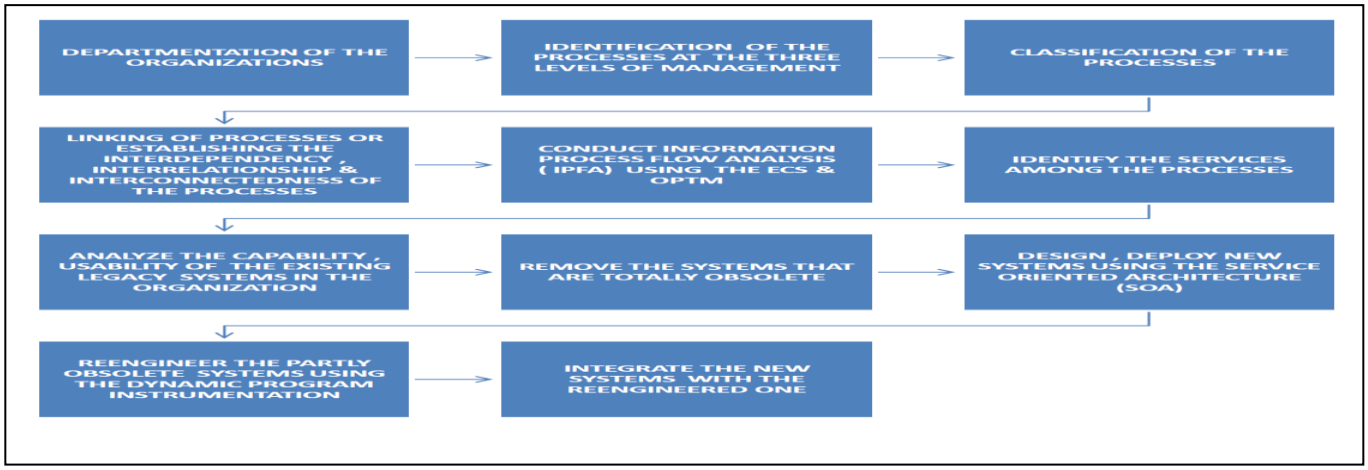

Fig. 1: The Object Oriented Knowledge Based (OKB) Model [Adopted from (Kundu, Ratha, Das, 2012)]

The last six steps of the OKB model is slightly different from their predecessors because these six steps together denote a particular process having four sequential phases Analyze , Design , Reengineer & Integrate and as per the name of these phases, the last six steps of the OKB model has been termed as ADRI approach [22]. Kundu , Ratha and Das's conceptualization of the ADRI approach is quite interesting as well as significant because this approach has been conceptualized with the single objective of combining the concept of services oriented architecture (SOA) and dynamic program analysis to produce a hybrid approach of system analysis, system design, system reengineering and system integration. The uniqueness of ADRI approach lies in the fact that it can perform all the four tasks of system analysis , system design , system reengineering and system integration in order to fulfill the ultimate objective of OKB model i.e. business process reengineering.This approach has six steps and four phases i.e. Analysis, Design, Reengineer and Integrate. All these steps are sequential in nature and these steps are needed to be performed in four phases (Ref. Fig. 2).

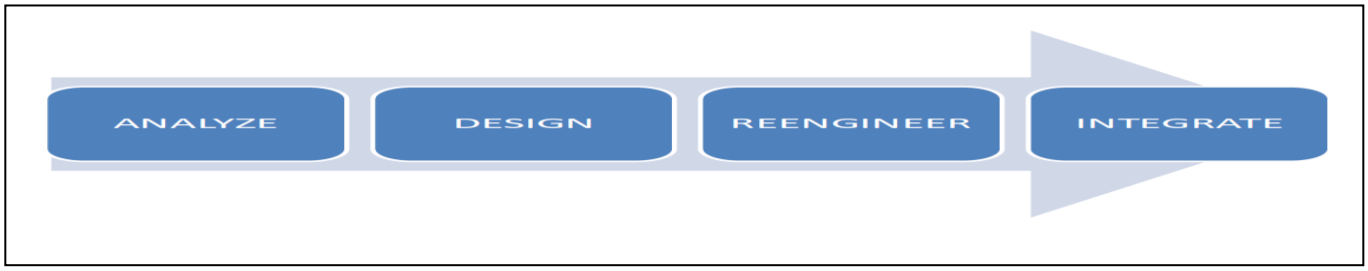

Fig. 2: The Sequential Steps of the ADRI Approach[Adopted from (Kundu, Ratha, Das, 2012)]

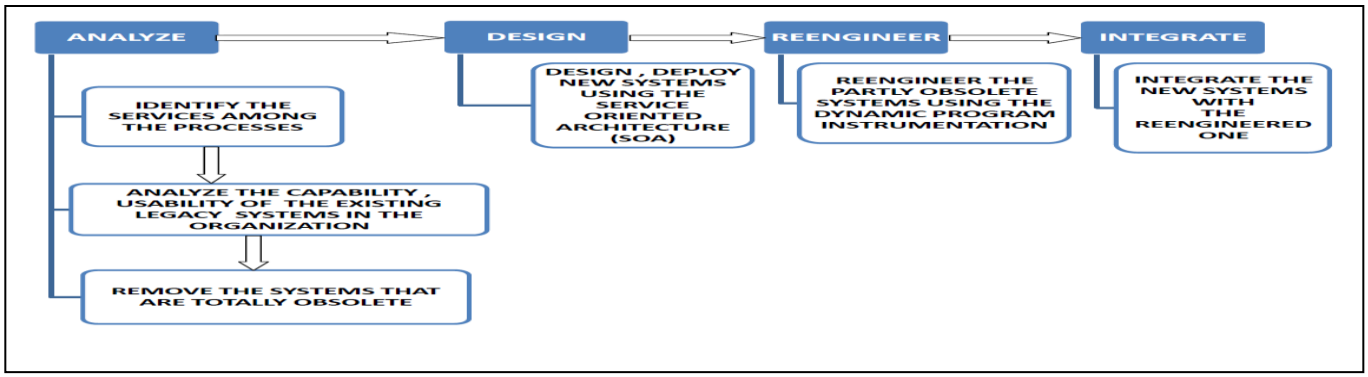

Fig. 3: Relationship between the phases and steps of ADRI approach[Adopted from (Kundu, Ratha, Das, 2012)]

The first phase is Analysis phase which has three steps and the rest of the three phases have one step each. Fig. 3 depicts the relationship between the phases and steps of ADRI approach. Kundu, Ratha and Das's work is seminal because their OKB model has far reaching significance in linking the concepts of SOA with BPR and Reverse Engineering (Ref. Fig.4).

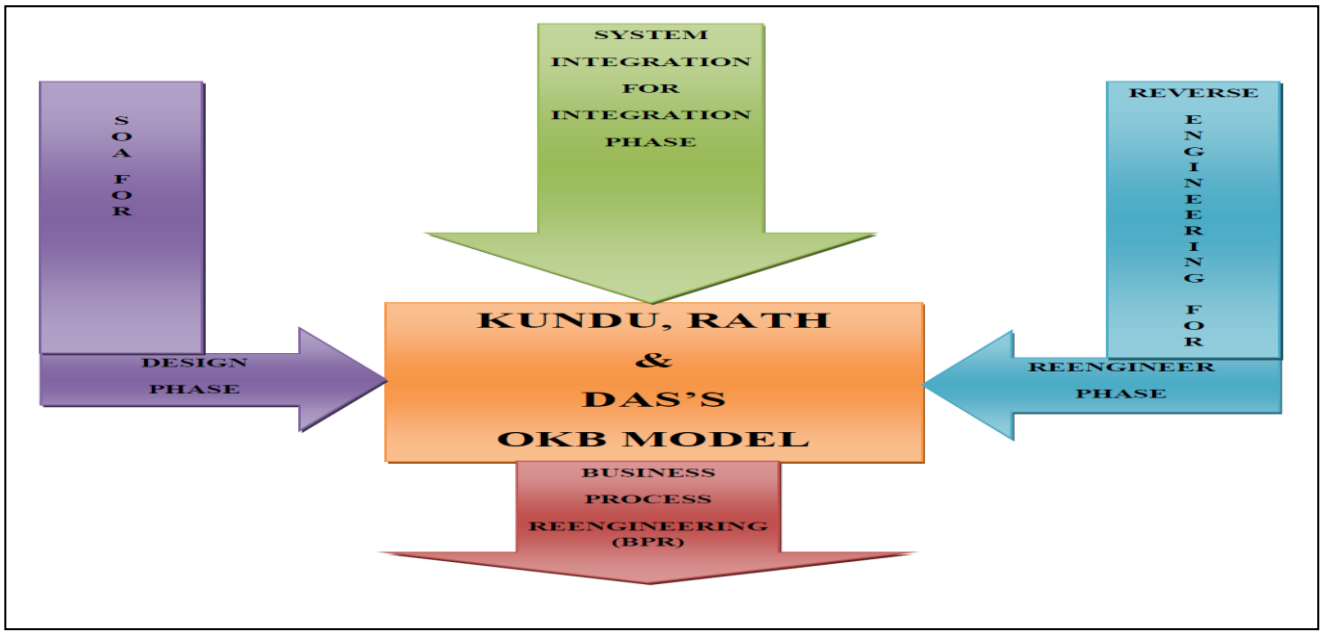

Fig. 4: The OKB Model links the concepts of SOA with BPR and Reverse Engineering

OKB model uses dynamic program instrumentation as a reverse engineering tool in the third phase of the ADRI approach i.e. the Reengineer phase. Again OKB model uses system integration tools in the fourth and final phase of the ADRI approach i.e. the Integrate phase. Also OKB model uses the concepts of SOA as a designing tool in the second phase of the ADRI approach i.e. the Design phase. So the OKB model encompasses the concepts of SOA, Reverse Engineering and System Integration through different phases of the ADRI approach (Ref. Fig.5).

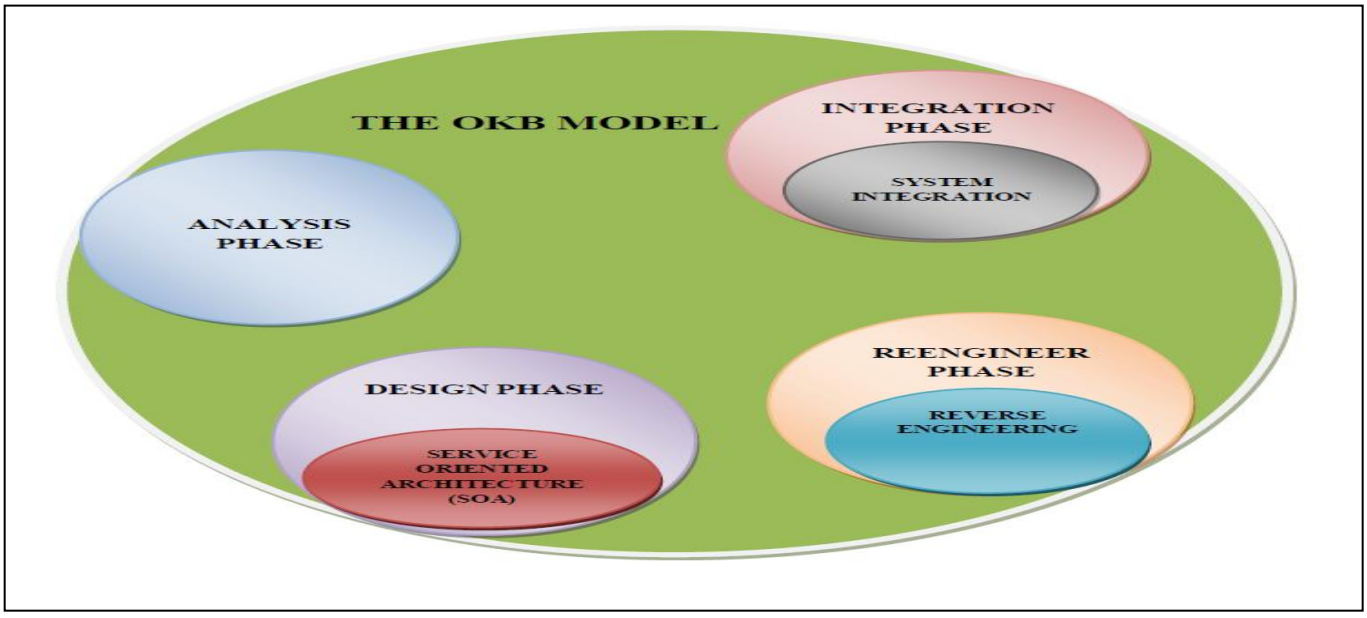

Fig. 5: The OKB Model encompasses the concepts of SOA, Reverse Engineering and System Integration through different phases of the ADRI approach

Das and Kundu developed a conceptual model for implementing a CRM model through SOA approach [22].Based on Kraf-zig, Banke and Slama's definition and representation of services in the SOA (Service Oriented Architecture) approach, Das and Kundu attempted to implement a CRM process by applying the concept of SOA([23],[21]). Earlier the researchers faced difficulty in using Web Services effectively because there was no mechanism to describe the web ser-vices and that's why in response to the urgent need of describing various characteristics of a web service unambiguously WSDL was created [24].In the literature, WSDL or Web Services Definition Language has been defined as a specification for describing the methods or data types of SOAP interface [25]. Ethan Cerami described the way of specifying services using WSDL in his book "Web Services Essentials: Distributed Applications with XML-RPC, SOAP, and UDDI & WSDL" [26]. Kundu, Das and Ratha extended the original idea of Das & Kundu's research on application of SOA in implementing a CRM process using the methods described by Ethan Cerami in his book "Web Services Essentials: Distributed Applications with XML-RPC, SOAP, and UDDI & WSDL" and thus developed WSDL specification of services for SOA based implementation of a CRM process[27] . Kundu, Das and Ratha's approach in developing WSDL specification of services for SOA based implementation of a CRM process is of utmost importance and significance because they proposed the roadmap and the WSDL codes to design services as per the SOA specifications which is the cornerstone of the design phase in the ADRI approach [22](Ref.Fig.6).

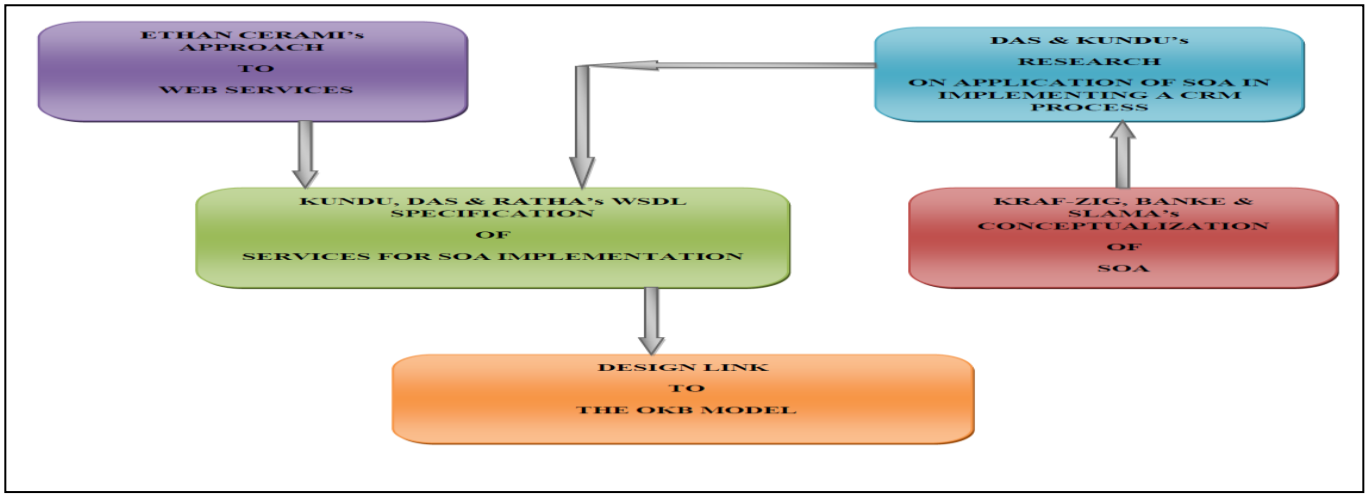

Fig. 6: Design Link to the OKB Model

In connection with the methods of software reverse engineering, Bach et al observe that dynamic program instrumentation can be implemented by performing the program instrumentation on the binary at runtime which will eliminate the need to modify or recompile the software application's source and it will also support the instrumentation of programs that dynamically generate code [28].

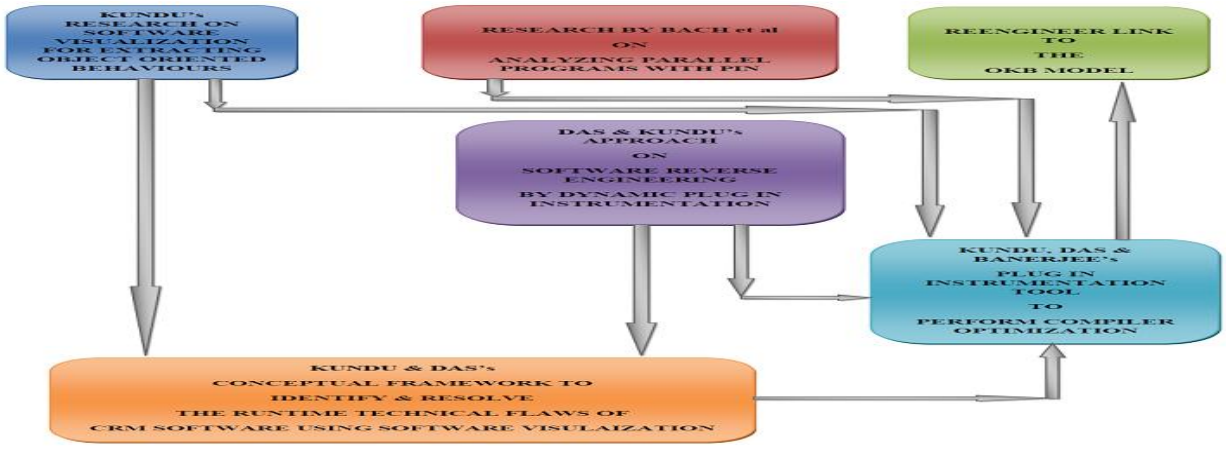

Fig. 7: Reengineer Link to the OKB Model

Based on Bach et al 's approach , Das and Kundu proposed a methodology of software reverse engineering in which an existing software can be reengineered by inserting a new module within the existing software by using dynamic plug-in instrumentation [29]. Kundu's research on software visualization for extracting object oriented behaviour shows that it is also possible to change instrumentation at any time during execution by modifying the application's binary image [30]. Based on Das and Kundu's approach on software reverse engineering by dynamic plug in instrumentation and using Kundu's software visualization approach, Kundu and Das advocated a conceptual framework to identify and resolve the runtime technical flaws of CRM software using software visualization [31]. Kundu and Das's framework serves two purposes; the primary purpose is to identify the runtime technical flaws of the CRM software and to take measures to solve the technical flaws using software visualization which can increase the level of program understanding and the secondary purpose is to use some optimization mechanisms based on death code elimination and code re-ordering approach to reduce the processing time of a CRM software package [31]. Lastly, Kundu, Das and Banerjee designed a plug in instrumentation tool based on all the previous research discussed before to perform compiler optimization and their approach has been used in the reengineer phase of the ADRI approach of the OKB Model to reengineer existing legacy CRM software ([22], [32])(Ref.Fig.7).Analysts and practitioners have developed techniques and methodologies on how to integrate legacy systems into SOA ([33]-[40]) and they believe that the process of integrating legacy systems into SOA starts with the identification of the main points of integration into the legacy mainframe. Under the OKB model, in the last phase of the ADRI approach i.e. the Integrate phase, system integration techniques have been discussed in details for proper integration of the old system with the new reengineered one.

6. Review of Gaps

As the cornerstone of this review paper is the OKB model of business process reengineering (BPR), it is quite natural that the analysis of the gaps among the BPR, SOA and reverse engineering will also involve the limitations and shortcomings of the OKB model. So, the limitations and shortcomings at different phases of the ADRI approach have been identified and also how these shortcomings can hamper the overall applicability and acceptability of the OKB model has been discussed. Kundu, Rath and Das rightly pointed out that as the OKB Model and the ADRI approach is based on the tools and techniques of SOA and reverse engineering, the inherent limitations of SOA and reverse engineering techniques will be present in this new model of business process engineering [22]. From this point of view it can be noted that as the concepts of SOA has been used in the Design phase and reverse engineering techniques have been used in the Reengineer phase of the ADRI approach, the gaps can be classified as Design level gaps and Reengineer level gaps.

6.1. Design Level Gaps

As Kundu, Das and Ratha's approach in developing WSDL specification of services for SOA based implementation of a CRM process provided the much needed roadmap and the WSDL codes to design services as per the SOA specifications which in turn is the cornerstone of the design phase in the ADRI approach [22], Kundu, Das and Ratha's observation on WSDL specification is of utmost importance. Kundu, Das and Ratha observe that the job of specifying the services through WSDL is more conceptual than technical in nature and that's why the description and specification of services through WSDL is not universal in nature and can vary according to the need of the programmer as well as per the clients' requirements [27]. Further, Cerami points out that the WSDL specification of services is the first step towards the SOA implementation of the conceptual CRM model and the actual implementation requires next level of research which is designing the WSDL Invocation Tools as per the IBM Web Services Invocation Framework (WSIF) [26]. Kundu, Rath and Das believe that though the applications of SOA have a promising future, many integration related issues are needed to be addressed while the transformation of legacy systems to SOA or migration to SOA platform [22]. Again, Caine and Hardman point out that as the legacy systems generally have proprietary data definitions; it creates the semantic discrepancy between legacy systems and other applications. [35]. Finally, the issue of standardization of SOA still remains a challenging issue and this is affecting the research in this field because in many cases the researchers find it difficult to assess the actual benefits of the SOA approach as an effective alternative to the existing legacy systems ([42], [43]).

6.2. Reengineering Level Gaps

As the application of dynamic program analysis and plug in instrumentation provide the key to the reengineer phase of the ADRI approach of the OKB model , so limitations and shortcomings of the dynamic program analysis and plug in instrumentation techniques form the basis of the reengineer level gaps. Das and Kundu believe that the dynamic program analysis and program instrumentation is still at the nascent stage and actual application in practice is an uphill task due to lack of efficient and faster program optimization techniques [29]. Kundu, Das and Banerjee point out that the working principle of the proposed plug-in Instrumentation is constrained by the limitation of execution coverage i.e. if the software program on which the proposed technique will be applied never reaches a particular point of execution, instrumentation tool at that point will not collect any data. They further observe that the application of some instrumentation techniques and tools may trigger a dramatic increase in execution time of particular software and this may limit the application of instrumentation tools to debugging contexts [32]. Researchers believe that the instrumentation technique proposed by Kundu and Das in their paper

titled "An Attempt to Analyze & Resolve the Pitfalls in CRM Software through Plug-In Instrumentation" is partially automated and fails to focus on dynamic load balancing issues ([31], [44], [45], [46]). Further, Kundu and Das point out that in their proposed instrumentation tool, security issues are another area where the proposed framework hasn't focused. They also observe that the challenges of developing a robust instrumentation technique on Windows operating systems lie in managing the kernel/application transitions, injecting the runtime agent into the process and isolating the instrumentation from the application [31]. Finally Kundu, Rath and Das recognize that their paper titled "Revisiting BPR: A New Framework & Model For Future" does not highlight anything about the architecture of the target CPU where the optimized CRM software is needed to be executed because compiler optimization also depends on the target CPU architecture that means on the no of registers usage, type of instruction set, pipelines and number of functional units present etc [22].

7. Conclusion & Future Directions

As this review paper is basically an elementary review encompassing the concepts of Service Oriented Architecture (SOA), Software Reverse Engineering and Business Process Reengineering (BPR) and identifying the linkages and gaps among these concepts from the point of view of the OKB model of BPR, this review paper should act as a base for future scientific works involving a more comprehensive review of literature.

References

[1] J.H. Manley. Information process flow analysis (IPFA) for reengineering manufacturing systems. Computers & Industrial Engineering, vol. 25, issue 1-4, pp. 275-278, September, 1993.

[2] K.C. Hoffman. Management of enterprise-wide systems integration programs. Journal of Systems Integration, vol. 3, issue 3-4, pp. 201 – 224, September 1993.

[3] J.H. Manley. Enterprise information system modeling for continuous improvement. Computers & Industrial Engineering, vol. 31, issue 1-2, pp. 273-276, October, 1996.

[4] H. Rozenfeld, A.F. Rentes, W. Konig. Workflow Modeling for Advanced Manufacturing Concepts. CIRP Annals - Manufacturing Technology, vol. 43, issue 1, pp. 385-388, 1994.

[5] R. P. Anjard. Re-engineering basics: one of three, new, special tools for management, quality and all other professionals. Microelectronics Reliability, vol. 36, issue 2, pp. 213-222.February, 1996.

[6] S. Jarzabek, W. L. Tok. Model-based support for business re-engineering. Information and Software Technology, vol. 38, issue 5, pp. 355-374. May, 1996.

[7] S. Kim, K. Jang. Designing performance analysis and IDEF0 for enterprise modeling in BPR. International Journal of Production Economics, vol. 76, issue 2, pp. 121-133. March 21, 2012.

[8] K. Mertins, R. Jochem. Architectures, methods and tools for enterprise engineering. International Journal of Production Economics, vol. 98, issue 2, pp. 179-188. November 18, 2005.

[9] E. J. Chikofsky and J. H. Cross II, Reverse engineering and design recovery: a taxonomy, IEEE Software, vol.7, issue. l, 13–17.1990.

[10] G.C.Harman and M.D.Penta. New frontiers of reverse engineering. FOSE'07 2007 Future of Software Engineering proceedings, pp.326-341, IEEE Computer Society, Washington DC, USA, 2007.

[11] E.J.Byrne. Software reverse engineering: a case study. Software: Practice and Experience, vol.21, issue.12, pp. 1349-1364.1991.

[12] E.Buss & J.Henshaw. Experiences in program understanding. Proceedings of the 1992 conference of the Centre for Advanced Studies on Collaborative research-Volume 1, pp. 157-189. IBM Press. November, 1992.

[13] G.Canfora, A.Cimitile & P.B.Lucarelli. Software maintenance. Handbook of Software Engineering and Knowledge Engineering, 1, 91-120.2002.

[14] R.R.Branco. Dynamic Program Analysis and Software Exploitation: From the crash to the exploit code. 2011. Retrieved on January 15, 2012, from http://www.troopers.de/wp-content/uploads/2011/04/TR11_Branco_Dynamic_program_analysis.pdf

[15] E.Cesar, A. Morajko, T. Margalef, J. Sorribes, A. Espinosa, E. Luque. Dynamic performance tuning supported by program specification. Performance oriented application development for distributed architectures, M.Gerndt, (Ed.), IOS Press, Amsterdam, pp. 35-44.2002.

[16] R.E.Ladner, R.Fortna, B.H. Nguyen. A Comparison of Cache Aware and Cache Oblivious Static Search Trees Using Program Instrumentation. Experimental algorithmics: from algorithm design to robust and efficient software, R.Fleischer, B. M. E. Moret, & E.M.Schmidt, (Eds.), Springer –Verlag, pp.78-92, Berlin, 2002.

[17] N.Bieberstein, S.Bose, M. Fiammante, K.Jones and R.Shah, Service Oriented Architecture Compass: Business Value, Planning and Enterprise Roadmap, Pearson Education, Inc, Upper Saddle River, 2006.

[18] J.P.Lawler, H.Howell- Barber, Service Oriented Architecture: SOA Strategy, Methodology and Technology, Auerbach Publications, 2007.

[19] N.Bieberstein, R.G.Laird, K.Jones and T.Mitra. Executing SOA: A practical Guide for the Service Oriented Architect, Pearson Education, Boston, 2008.

[20] N.M.Josuttis. SOA in Practice: The Art of Distributed System Design, O'Reilly Media Inc, Sebastopol, 2007.

[21] D.Krafzig, K.Banke, D. Slama. Enterprise SOA: Service Oriented Architecture: Best Practices, Pearson Education, Upper Saddle River, 2005.

[22] P.Kundu, B.K.Ratha and D.Das. Revisiting BPR: A New Framework & Model for Future, International Journal of Engineering Research and Technology, Vol. 1, Issue 8. October2012. Retrieved on January 13, 2012 from http://ssrn.com/abstract=2169913

[23] D.Das, P.Kundu. Application of Service Oriented Architecture (SOA) in Implementing a CRM Process: A Conceptual Approach , Research and Higher Education in Computer Science and Information Technology, S. S. Sau and A. D. Gupta,(Eds.),SM,pp.148-152.Kolkata,2012. Retrieved on 10th April 2012 from http://ssrn.com/abstract=2016277

[24] A.D.Nghiem.IT Web Services: A Roadmap for the Enterprise, Pearson Education, Inc., Upper Saddle River, 2003.

[25] W.Iverson. Real World Web Services, O'Reilly Media, Inc., Sebastopol, 2004.

[26] E.Cerami. Web Services Essentials: Distributed Applications with XML-RPC, SOAP, UDDI & WSDL, O'Reilly & Associates Inc., Sebastopol, 2012.

[27] P.Kundu, D.Das, B.K.Ratha. WSDL Specification of Services for Service Oriented Architecture (SOA) Based Implementation of A CRM Process. International Journal of Scientific & Engineering Research, vol. 3, issue 10, October2012. Retrieved on January 13, 2013 from http://ssrn.com/abstract=2159799

[28] M.Bach, M.Charney, R.Cohn, E.Demikhovsky, T.Devor, K.Hazelwood, A.Jaleel, C.K.Luk, G.Lyons, H.Patil, A.Tal. Analyzing Parallel Programs with Pin. 2010. Retrieved January 15, 2012, from http://www.cs.virginia.edu/kim/docs/ieeeComputer10.pdf

[29] D.Das, P.Kundu. Applications of Dynamic Program Analysis in a CRM Process: A Futuristic Concept. International Journal of Scientific and Research Publications, vol.2, issue. 4, pp. 306-318. Apr.2012. Retrieved on January 13, 2013 from http://ssrn.com/abstract=2042588

[30] P.Kundu. Visualization of Program for Extracting Object Oriented Behaviour & Optimization: A Futuristic Approach. Research and Higher Education in Computer Science and Information Technology, S. S. Sau and A. D. Gupta, (Eds.), SM, pp.108-112. Kolkata, 2012.

[31] P.Kundu, D.Das. An Attempt to Analyze & Resolve the Pitfalls in CRM Software through Plug-In Instrumentation. International Journal of Scientific and Research Publications, vol. 2, Issue 5, pp. 313 – 320.May, 2012. Retrieved on January 13, 2013 from http://ssrn.com/abstract=2051886

[32] P.Kundu, D.Das, A.Banerjee. Plug-In Instrumentation: A futuristic ICT approach for teaching the Runtime Behaviour of Software. U.G.C Sponsored Seminar on I.C.T in Higher Education: Opportunities & Challenges in the 21st Century Proceedings, SPS Education India Pvt. Ltd., pp. 24-27. Kolkata, 2012. Retrieved on January 13, 2013 from http://ssrn.com/abstract=2177249

[33] A.Papkov. Develop a migration strategy from a legacy enterprise IT infrastructure to SOA-based enterprise architecture. 2005. Retrieved on April 13, 2012 from http://www.ibm.com/developerworks/webservices/library/ws-migrate2soa/

[34] K.Channabasavaiah, E.Tuggle, Jr., K. Holley. Migrating to a service-oriented architecture. 2003. Retrieved on April 13, 2012 from http://www.ibm.com/developerworks/webservices/library/ws-migratesoa/

[35] J.Caine, J. Hardman. Design strategies for legacy system involvement in SOA solutions: Understand the challenges of business transformation using SOA in legacy environments. 2007. Retrieved on April 13, 2012 from http://www.ibm.com/developerworks/webservices/library/ws-soa-legacy/

[36] C.Lawrence. Adapting legacy systems for SOA: Finding new business value in older applications. 2007. Retrieved on April 13, 2012 from http://www.ibm.com/developerworks/webservices/library/ws-soa-adaptleg/

[37] C. Nott. Patterns: Using Business Service Choreography in Conjunction with an Enterprise Service Bus.2004. Retrieved on April 13, 2012 from http://www.redbooks.ibm.com/redpapers/pdfs/redp3908.pdf

[38] M.Keen et al. Patterns: Implementing an SOA Using an Enterprise Service Bus. 2004. Retrieved on April 13, 2012 from http://www.redbooks.ibm.com/redbooks/pdfs/sg246346.pdf

[39] T. Laszewski, J. Williamson. SOA Integration in the Legacy Environment. 2008. Retrieved on April 13, 2012 from http://www.oracle.com/technetwork/articles/laszewski-williamson-soa-legacy-082027.html

[40] M.A.A.Sheikh, H.A. Aboalsamh, A. Albarrak. Migration of legacy applications and services to Service-Oriented Architecture (SOA). International Conference and Workshop on Current Trends in Information Technology (CTIT) Proceedings, pp. 137 – 142, 2011.

[41] W. Xiaofeng, S.X.K.Hu, E. Haq, H. Garton. Integrating Legacy Systems within The Service-oriented Architecture. IEEE Power Engineering Society General Meeting 2007 Proceedings, pp.1 – 7.2007.

[42] J.Erickson and K.Siau. Web Services, Service Oriented Computing & Service Oriented Architecture. Principle Advancements in Database Management Technologies: New Applications, K.Siau and J.Erickson, Eds. Hershey, PA: Information Science Reference, pp.176-187, 2010.

[43] P.A.Baer, Platform Independent Development of Robot Communication Software. Zugl, Kassel: Kassel University Press, 2008.

[44] C.Xu & F.C.M.Lau, *Load Balancing In Parallel Computers: Theory & Practice*. Norwell, MA: Kluwer Academic Publishers, 1997.

[45] R.V.Dorneles, R.L.Rizzi, A.L.Martinotto, D.Picinin Jr., P.O.A.Navaux and T.A.Diverio, ―Parallel Computational Model with Dynamic Load Balancing in PC Clusters. VECPAR 2004, LNCS 3402, M. Dayde et al., Eds. Berlin Heidelberg: Springer – Verlag, pp. 468-479, 2005.

[46] K.S.Chatrapati, J.U.Rekha and A.V.Babu, ―Recursive Competitive Equilibrium Approach for Dynamic Load Balancing a Distributed System. ICDCIT 2011, LNCS 6536, R. Natarajan and A.Ojo et al., Eds. Berlin Heidelberg: Springer – Verlag, pp. 162-174, 2011.