JOB ADVERT

JOB TITTLE Software Dev Manager

JOB PURPOSE Intertoll Africa invites suitably qualified candidates to apply for the permanent position of Software Development Manager. The successful candidate will take full accountability for designing, developing, testing, integrating and interfacing software systems on time and to client's requirements across various systems (e.g. tolling). This includes creating, supporting and/or customizing solutions for clients, as well as managing a development team.

QUALIFICATION S: B-Tech in IT/ Computer Science / B. Comm informatics

EXPERIENCE: 7 years+ experience in programming, project management and software implementation. (.Net, C#, T-SQL, SQL Server, Visual Studio, etc.)At least 2 years’ experience in database design and working with ORM's (especially Entity Framework). Experience of Agile methodologies. Experience in JavaScript and client-side libraries like JQuery and KnockoutJS. Experience with object-orientated designs and techniques. Good working experience with Team Foundation Server.

DUTIES & RESPONSIBILITIES Software development. Software requirements analysis. Software coding and implementation. Software Unit testing, integration and documentation. SHE requirements compliance, i.e. ISO 9001, 14001 and 18001. Achievement of project deliverables. Post-implementation client support and maintenance. Accurate, timely reporting. Management of adherence to IT governance, risk and compliance standards. Stakeholder engagement. Team leadership, management and mentoring. Teamwork, self-management and alignment with group values. Analyse workload and availability of personnel and resources. Work with business analysists to create technical architectural design documents that meet client requirements.

SKILLS AND KNOWLEDGE Knowledge of the organisational structures. Systems requirement specifications (SRS). Interface requirements specifications (IRS). Software Test Description (STD). Interface Design Description (IDD). Software Design Description (SDD). System Compliance Certification. Understand functional and technical documentation needs of customers.

ADDITIONAL INFORMATION Strong analytical skills and attention to detail. Excellent Problem-solving ability. Excellent communication skills, both verbal and written. Ability to work in a highly pressurised environment and deliver on tight deadlines. Ability to work independently. Provide technical leadership to the team on design, development and implementation of various systems. Project coordination and management skills. Support and guide the Software Engineers on project specification deadlines. Ensure design development is in line with best practice.

 Software engineering is the application of engineering to the development of software in a systematic method.  
Definitions   
Notable definitions of software engineering include:   
"the systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software"—The Bureau of Labor Statistics—IEEE Systems and software engineering - Vocabulary  
"The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software"—IEEE Standard Glossary of Software Engineering Terminology  
"an engineering discipline that is concerned with all aspects of software production"—Ian Sommerville  
"the establishment and use of sound engineering principles in order to economically obtain software that is reliable and works efficiently on real machines"—Fritz Bauer  
The term has also been used less formally:  
as the informal contemporary term for the broad range of activities that were formerly called computer programming and systems analysis;  
as the broad term for all aspects of the practice of computer programming, as opposed to the theory of computer programming, which is called computer science;  
as the term embodying the advocacy of a specific approach to computer programming, one that urges that it be treated as an engineering discipline rather than an art or a craft, and advocates the codification of recommended practices.  
History   
When the first digital computers appeared in the early 1940s, the instructions to make them operate were wired into the machine. Practitioners quickly realized that this design was not flexible and came up with the "stored program architecture" or von Neumann architecture. Thus the division between "hardware" and "software" began with abstraction being used to deal with the complexity of computing.  
Programming languages started to appear in the early 1950s and this was also another major step in abstraction. Major languages such as Fortran, ALGOL, and COBOL were released in the late 1950s to deal with scientific, algorithmic, and business problems respectively. David Parnas introduced the key concept of modularity and information hiding in 1972 to help programmers deal with the ever-increasing complexity of software systems.  
The origins of the term "software engineering" have been attributed to various sources. The term "software engineering" appeared in a list of services offered by companies in the June 1965 issue of and was used more formally in the August 1966 issue of Communications of the ACM “letter to the ACM membership” by the ACM President Anthony A. Oettinger;, it is also associated with the title of a NATO conference in 1968 by Professor F.L. Bauer, the first conference on software engineering.. At the time there was perceived to be a "software crisis".  
In 1984, the Software Engineering Institute was established as a federally funded research and development center headquartered on the campus of Carnegie Mellon University in Pittsburgh, Pennsylvania, United States. Watts Humphrey founded the SEI Software Process Program, aimed at understanding and managing the software engineering process. The Process Maturity Levels introduced would become the Capability Maturity Model Integration for Development, which has defined how the US Government evaluates the abilities of a software development team.  
Modern, generally accepted best-practices for software engineering have been collected by the ISO/IEC JTC 1/SC 7 subcommittee and published as the Software Engineering Body of Knowledge .  
Subdisciplines   
Software engineering can be divided into sub-disciplines. Some of them are:  
Software requirements  
Many software engineers enter the profession by obtaining a university degree or training at a vocational school. One standard international curriculum for undergraduate software engineering degrees was defined by the Joint Task Force on Computing Curricula of the IEEE Computer Society and the Association for Computing Machinery, and updated in 2014. A number of universities have Software Engineering degree programs;, there were 244 Campus Bachelor of Software Engineering programs, 70 Online programs, 230 Masters-level programs, 41 Doctorate-level programs, and 69 Certificate-level programs in the United States.  
In addition to university education, many companies sponsor internships for students wishing to pursue careers in information technology. These internships can introduce the student to interesting real-world tasks that typical software engineers encounter every day. Similar experience can be gained through military service in software engineering.  
Profession   
Legal requirements for the licensing or certification of professional software engineers vary around the world. In the UK, there is no licensing or legal requirement to assume or use the job title Software Engineer. In some areas of Canada, such as Alberta, British Columbia, Ontario, and Quebec, software engineers can hold the Professional Engineer designation and/or the Information Systems Professional designation. In Canada, there is a legal requirement to have P.Eng when one wants to use the title "engineer" or practice "software engineering". In Europe, Software Engineers can obtain the European Engineer professional title.  
The United States, since 2013, has offered an NCEES Professional Engineer exam for Software Engineering, thereby allowing Software Engineers to be licensed and recognized. Mandatory licensing is currently still largely debated, and perceived as controversial. In some parts of the US such as Texas, the use of the term Engineer is regulated by law and reserved only for use by individuals who have a Professional Engineer license.  
The IEEE Computer Society and the ACM, the two main US-based professional organizations of software engineering, publish guides to the profession of software engineering. The IEEE's Guide to the Software Engineering Body of Knowledge - 2004 Version, or SWEBOK, defines the field and describes the knowledge the IEEE expects a practicing software engineer to have. The most current SWEBOK v3 is an updated version and was released in 2014. The IEEE also promulgates a "Software Engineering Code of Ethics".  
Employment   
In November 2004, the U. S. Bureau of Labor Statistics counted 760,840 software engineers holding jobs in the U.S.; in the same time period there were some 1.4 million practitioners employed in the U.S. in all other engineering disciplines combined. Due to its relative newness as a field of study, formal education in software engineering is often taught as part of a computer science curriculum, and many software engineers hold computer science degrees and have no engineering background whatsoever.  
Many software engineers work as employees or contractors. Software engineers work with businesses, government agencies, and non-profit organizations. Some software engineers work for themselves as freelancers. Some organizations have specialists to perform each of the tasks in the software development process. Other organizations require software engineers to do many or all of them. In large projects, people may specialize in only one role. In small projects, people may fill several or all roles at the same time. Specializations include: in industry and in academia .  
Most software engineers and programmers work 40 hours a week, but about 15 percent of software engineers and 11 percent of programmers worked more than 50 hours a week in 2008. Injuries in these occupations are rare. However, like other workers who spend long periods in front of a computer terminal typing at a keyboard, engineers and programmers are susceptible to eyestrain, back discomfort, and hand and wrist problems such as carpal tunnel syndrome.  
Certification   
The Software Engineering Institute offers certifications on specific topics like security, process improvement and software architecture. IBM, Microsoft and other companies also sponsor their own certification examinations. Many IT certification programs are oriented toward specific technologies, and managed by the vendors of these technologies. These certification programs are tailored to the institutions that would employ people who use these technologies.  
Broader certification of general software engineering skills is available through various professional societies., the IEEE had certified over 575 software professionals as a Certified Software Development Professional . In 2008 they added an entry-level certification known as the Certified Software Development Associate . The ACM had a professional certification program in the early 1980s, which was discontinued due to lack of interest. The ACM examined the possibility of professional certification of software engineers in the late 1990s, but eventually decided that such certification was inappropriate for the professional industrial practice of software engineering.  
In the U.K. the British Computer Society has developed a legally recognized professional certification called Chartered IT Professional, available to fully qualified members . Software engineers may be eligible for membership of the Institution of Engineering and Technology and so qualify for Chartered Engineer status. In Canada the Canadian Information Processing Society has developed a legally recognized professional certification called Information Systems Professional . In Ontario, Canada, Software Engineers who graduate from a Canadian Engineering Accreditation Board accredited program, successfully complete PEO's Professional Practice Examination and have at least 48 months of acceptable engineering experience are eligible to be licensed through the Professional Engineers Ontario and can become Professional Engineers P.Eng. The PEO does not recognize any online or distance education however; and does not consider Computer Science programs to be equivalent to software engineering programs despite the tremendous overlap between the two. This has sparked controversy and a certification war. It has also held the number of P.Eng holders for the profession exceptionally low. The vast majority of working professionals in the field hold a degree in CS, not SE. Given the difficult certification path for holders of non-SE degrees, most never bother to pursue the license.  
Impact of globalization   
The initial impact of outsourcing, and the relatively lower cost of international human resources in developing third world countries led to a massive migration of software development activities from corporations in North America and Europe to India and later: China, Russia, and other developing countries. This approach had some flaws, mainly the distance / timezone difference that prevented human interaction between clients and developers and the massive job transfer. This had a negative impact on many aspects of the software engineering profession. For example, some students in the developed world avoid education related to software engineering because of the fear of offshore outsourcing and of being displaced by foreign visa workers. Although statistics do not currently show a threat to software engineering itself; a related career, computer programming does appear to have been affected. Nevertheless, the ability to smartly leverage offshore and near-shore resources via the follow-the-sun workflow has improved the overall operational capability of many organizations. When North Americans are leaving work, Asians are just arriving to work. When Asians are leaving work, Europeans are arriving to work. This provides a continuous ability to have human oversight on business-critical processes 24 hours per day, without paying overtime compensation or disrupting a key human resource, sleep patterns.  
While global outsourcing has several advantages, global - and generally distributed - development can run into serious difficulties resulting from the distance between developers. This is due to the key elements of this type of distance that have been identified as geographical, temporal, cultural and communication . Research has been carried out in the area of global software development over the last 15 years and an extensive body of relevant work published that highlights the benefits and problems associated with the complex activity. As with other aspects of software engineering research is ongoing in this and related areas.  
Related fields   
Software engineering is a direct sub-field of engineering and has an overlap with computer science and management science . It is also considered a part of overall systems engineering.  
Computer Science   
In general, software engineering focuses more on techniques for the application of software development in industry, while computer science focuses more on algorithms and theory.  
Controversy   
Criticism   
Software engineering sees its practitioners as individuals who follow well-defined engineering approaches to problem-solving. These approaches are specified in various software engineering books and research papers, always with the connotations of predictability, precision, mitigated risk and professionalism. This perspective has led to calls for licensing, certification and codified bodies of knowledge as mechanisms for spreading the engineering knowledge and maturing the field.  
Software craftsmanship has been proposed by a body of software developers as an alternative that emphasizes the coding skills and accountability of the software developers themselves without professionalism or any prescribed curriculum leading to ad-hoc problem-solving without engineering . extends the Agile Software Manifesto and draws a metaphor between modern software development and the apprenticeship model of medieval Europe.  
Software engineering extends engineering and draws on the engineering model, i.e. engineering process, engineering project management, engineering requirements, engineering design, engineering construction, and engineering validation. The concept is so new that it is rarely understood, and it is widely misinterpreted, including in software engineering textbooks, papers, and among the communities of programmers and crafters.  
One of the core issues in software engineering is that its approaches are not empirical enough because a real-world validation of approaches is usually absent, or very limited and hence software engineering is often misinterpreted as feasible only in a "theoretical environment."  
Edsger Dijkstra, the founder of many of the concepts used within software development today, refuted the idea of "software engineering" up until his death in 2002, arguing that those terms were poor analogies for what  
he called the "radical novelty" of computer science:  
See also   
Bachelor of Science in Information Technology  
Bachelor of Software Engineering  
List of software engineering conferences  
List of software engineering publications  
Software craftsmanship  
Software Engineering Institute  
Notes   
References   
Further reading   
External links   
OpenSDLC.org the integrated Creative Commons SDLC  
Carnegie Mellon  
Software Engineering Society  
  
  
Bibliography:  
Wikipedia  
@baygross