Design & Application of an Assessment of Coding Competence

Technical Manual*

Reviewed & Prepared by

R. Chris Steilberg, PhD

* This content of this document represent work in progress as of April, 2018. All information contained herein accurately presents the design, development and reliable psychometric properties of the Filtered. Updates to this document will be made as additional, dependable data concerning its psychometric properties are obtained. Such data may inform revisions to the assessment. If so, any and all revisions will be documented and made available as appropriate.

CONTENTS

INTRODUCTION

The Filtered

Description. The Filtered is a fully-automated, web-based system that incorporates artificial intelligence (AI) to support the most accurate, efficient and standardized assessment of candidates for highly technical jobs, specifically those pertaining to software engineering.

Content. The Filtered is designed to assess two capability domains: 1) highly specialized coding skills and 2) behavioral competencies. Content for both of these categories is available from Filtered Ai's extensive library of coding items and competency-based interview questions. In addition, clients have the option to include their own content for either category domain.

Delivery. The Filtered is a fully-automated, self-administering online assessment compatible with virtually any technology commonly available to and used by software engineers. Applicants are video-recorded by their technology throughout the entire assessment process to minimize issues concerning test security and cheating.

Applicants may complete the assessment at their convenience as long as they complete the Filtered within timeframes established by the hiring party. This feature eliminates the burden of scheduling "bottlenecks" inherent to live interviews. Moreover, access to the Filtered can be provided directly within the corresponding job's online posting. Consequently, the Filtered capitalizes on two of the most powerful and prevalent influences on behavior: 1) response immediacy/availability (i.e., the Filtered is incorporated within the job posting thereby making it clearly accessible) and 2) motivation (i.e., the Filtered is available to active {motivated} job seekers).

Background

Labor Market. Challenges in the recruitment and selection of software engineers[1] are significant and carry considerable risk. According to the Bureau of Labor Statistics, the demand for software engineers is estimated to grow "much faster than average" at a rate of approximately 31% over 10 years beginning 2016 (median growth rate is 7%)[2]. Ranking amongst the fastest growing jobs, the vast majority of programmers in the workforce have 4-year college degrees or higher and command median salaries over six-figures[3]. Consequently, three phenomena have emerged in this labor market: 1) fierce competition for talent, 2) more decision power to the candidate and, 3) relaxed standards for selection.

[1] The terms "software engineer," "programmer," "coders" in this document are variably used as specific jobs within the job family, "code designers." They are not exhaustive and are not supported by empirical analyses. They are among a larger set of jobs frequently cited as "comparable" in O*Net.
[2] Occupational Outlook Handbook (Jan 2018), a publication of The Bureau of Labor Statistics.
[3] Onetonline.org (Software Developers, 2017 data)

Research providing empirical evidence for the means and extent to which these phenomena actually influence assessment decisions and selection outcomes is scant and primarily anecdotal. But, these same phenomena adhere to well documented principles of market dynamics and are evident to anyone in the market for valued and rare products. In this case, the products also know this and make the "purchasing" decision even more challenging.

Regardless of resources, organization reputation or the relative decision power of candidates, the criticality of accurate selection remains high. Larger organizations may be able to "absorb" hiring mistakes, but this may lead to rapid, and costly, errors. Smaller organizations that do a poor job assessing candidates risk talent dilution by taking on a sub-par contributor. Finally, when power shifts toward the candidate, desperation can set in with the result of over bidding. It pays to know how much something, or someone is worth before negotiating.

And these are just the most obvious challenges to specialist selection. More concerning are the subtle errors, unaware to the hiring party, that lead to bad hiring decisions.

Selection Bias. Assessment is never completely error-proof. For the most part, these errors are unintentional; hiring parties want to "get it right." But when assessment is systematically influenced by inaccurate cognitive processes, selection bias is likely. One form of selection bias is especially risky as it pertains to hiring for highly specialized candidates, even more so when the specialization is knowledge-based.

Across the spectrum of competency assessment, procedural knowledge (applied knowledge) is among the easier to identify. Since procedural knowledge (viz., coding) is well-suited to the format of resumes and is relatively easy to verify (should there be any doubt), it is a prime candidate for error-prone assumptions regarding software engineers.

"Certified in JQuerry, C++, Javascript," or "MS in Computer Science" from a well-known institution may be all a hiring manager needs to hear or see in order to "check the box" of a clear skill. It's precisely in this unique case, selection errors are more likely due to the ease with which a hiring manager may generate expectations based on his or her *opinion* as informed by declarative (and, presumably definitive) resume content.

Confirmation Bias. Errors of the type previously mentioned constitute "confirmation bias." Confirmation biases derail objective evaluations by systematically skewing judgment based on previously formed attributions. These attributions elicit unsupported, and frequently compounding judgments. Once primed, the confirmation bias creates "selective attention" to identify, "confirm" and reinforce a priori beliefs.
The Filtered guards against confirmation bias by presenting applicant demographic detail at the end of the assessment process. As such, tests have been taken and scored and interviews have been delivered and evaluated by the hiring party before resumes are revealed. This practice insures that all applicants are tested according to identical standards. Although not published

here, Filtered Ai has seen significant improvements in the representation of female hires as a result of using the Filtered as compared to organizations' previous methods.

Finally, industry reports suggest that IT leadership have a skill gap relative to coders. This isn't surprising at all, but it further obfuscates accurate assessment of skills without expert support. It is also likely to contribute to the overestimate of technical skill relative to other job—relevant competencies.

Summary

A summary of challenges to the accurate assessment of software engineers follows.

1. These are high stakes decisions that can force uninformed decisions.
2. The specific and rapidly evolving skill sets in software development have been shown to create a knowledge gap between organization levels. Absent understanding of code content, leaders are at risk of superficial evaluation, frequently driven by labels.
3. A variety of factors (critical skill sets, knowledge gaps between hiring party and candidate) can contribute to overlooking other critical competencies.
4. Conditions for confirmation bias are ripe. Past experience, education or certifications can all contribute to false assumptions of candidate competence that grant a "hall pass" to more objective assessment.

The Filtered dramatically reduces these and other risks unique, but inherent to hiring software engineers.

TECHNICAL

DESIGN

The Basic Structure of The Filtered

Two Component Approach. Demonstrated competence is a key principle of the Filtered approach to the accurate assessment of programming candidates. As such, work samples are the primary means of gauging the technical skills and general competencies of software engineer candidates. In the case of coding, the work samples comprise real time programing challenges to solve. As for the more subjective general competencies, evidence of competence is gathered by structured interviews.

Technical/Functional Expertise

Content Generation

Background. Computer coding has evolved substantially from the era of mainframes and punch cards. At that time, coding was limited in terms of its variance. Over time, a multitude of software systems and applications have emerged. Likewise, computer coding, the specific syntax underlying any application, has evolved in quality and quantity with hundreds of different coding "languages."

Highest Demand Coding Skills. One of the first decisions that must be made when designing content for code-based simulations is which ones to include. New coding languages are emerging at an increasingly rapid rate, but older forms of code continue to have their place in applications. It's not enough to learn the newer coding skills, good programmers accumulate different skills throughout their career. To make matters even more complex, the variety of packages in the burgeoning market of software systems means that codes prevalent in one department of an organization may be unheard of in another. This effect is true across organizations as well.

Filtered Ai leveraged automated web crawling to identify the coding languages in greatest demand as represented by over 100,000 job postings and descriptions. These skills were ranked according to both frequency of mention and importance to the most jobs. This priority scheme created a development plan to insure optimal representation of key codes.

Levels of Expertise. Beyond covering the *breadth* of the content domain for software engineers (i.e., different programing languages), the Filtered assesses the expertise of programmers according to their *level* of skill. As such, three separate assessments, each with their own item-bank, were designed to most appropriately assess competence for the three levels of expertise. The three test levels and corresponding populations for use according to experience include:

A) Intermediate – candidates with no experience beyond education to 2 years' experience
B) Advanced – candidates with 3 – 5 years' experience
C) Master – candidates with more than 5 years' experience

Item Banks by Level. In order to accurately (and compassionately) assess differences among candidates within a given skill level, item difficulties were clustered about the mean competence of engineers reflective of that level. Differences between levels were insured by creating items more alike in terms of difficulty *within* a level than they were *between* skill levels. This approach to item design and allocation to levels minimized the likelihood of a very high scoring candidate in a lower level assessment actually being more skilled than a lower scoring applicant in the next level up.

Performance vs. Potential. The coding test of the Filtered assesses predictable job performance. The test is actually a work sample composed of items consistent with actual job tasks and demands. Therefore, scores can only be used to accurately predict performance for a specific job, not potential for advancement. This does NOT mean that a candidate or a new hire can't advance to higher levels (i.e., has potential), but this cannot, and should not, be an assumption based on test coding skill. Candidate potential is more amenable to evaluation in the interview component of the assessment. Various competencies may be indicative of potential (e.g., learning agility).

The Filtered Coding Library. Thirty-five different coding languages and corresponding assessments now comprise a growing library of The Filtered coding skills assessments. Each of these assessments has been carefully designed to support the valid, fair, legal selection of candidates for jobs in the software engineering job family[4]. (Details regarding validity are subsequently presented). More coding languages continue to be added to the Filtered library as the consumer appetite for new technology drives organizations to constant innovation oftentimes requiring change all the way to the cutting line of code.

Assessment Approach

Code-based Test Items. While evidence demonstrating the strength of the work-sample approach for selection is extensive and compelling, documented research specific to software engineers is limited despite a multitude of factors to support its appropriateness for precisely these jobs[5]. This is likely due to three factors:

1. Computer coding content and web-based assessment are relatively new to the field of psychometric testing.

[4] Local validation is required of any assessment to accurately gauge validity of use and to insure full compliance with Uniform Guidelines and EEOC.
[5] But see M. Lynch for an exceptional affirmation of the value of work samples in coding assessment, Lynch, M. (2017) "Testing Job Applicants for CNC Proficiency." *Modern Machine Shop.* pp 62-64.

2. Tests of declarative (know it) and procedural (do it) knowledge offer very little value to academia.
3. Most applied examples are proprietary; deliberately concealed from the public.

The second of these factors is likely to be the main reason for limited research; the third for limited publication. Few would be surprised to learn that this coding skill is well assessed via computer-based assessment. In fact, it's so apparent, some are likely to ask, "why bother at all?" Herein lies the impetus of confirmation bias, "if it's so easy to prove, is it really necessary to test for it?"

Scoring. In order to reflect criteria characteristic of programmer performance at work, scoring of the technical component of the Filtered incorporates aspects of both efficiency and efficacy in terms of:

A) How quickly items are solved (Time to solve)
B) How efficiently the code is composed (Memory used)
C) How effectively the item is solved (Accuracy of solution)

The issue of accuracy goes beyond "right/wrong." Items require a sequence of steps. Credit is given to reflect partial accuracy of each solution produced. Thus, a candidate is not over-penalized if the principle components of the solution are evident even if the ultimate solution is incorrect.

Ten items, or coding challenges, are included for every Filtered coding test. Scores are summed across items to determine a candidate's overall score with higher scores reflecting higher skill in the tested coding language. These scores are normed against all other applicants completing the assessment to produce their rank as a percentile relative to the sample population.

Guidance - Not Recommendation. Selection dynamics vary considerably depending on factors including, but not limited to, hiring organization demand, market demand, candidate supply, affordability, etc. Filtered cannot, and does not, know what factors are in play at any time for any organization. As a result, no recommendations are made regarding hiring decisions.

The design of the coding test does provide an objective assessment of current skill level in the various codes assessed relative to other candidates. As such, the Filtered produces, scores and norms coding assessments designed, as recommended, according to principles of content validity (see Validity, below). This does NOT mean that criterion-related validity is a non-issue. Criterion-referenced validity is recommended for every application of The Filtered. These studies must be conducted at the level of the organization, not the assessment, in order to most appropriately document and defend the valid use of the Filtered for the purpose of selection.

General Competencies

Content

Task-oriented Analyses. Technical knowledge is necessary but insufficient in order to be maximally effective in any job, including programmers. Formal job analyses of coding-related jobs and incumbents routinely identify: communication, logical reasoning, mathematical fluency and information ordering as competencies that are very important to job performance[6].

Worker-oriented Analyses. A second approach was used to generate items that task-oriented job analyses don't always capture due to their primary focus on knowledge, skills and abilities. A "worker-oriented" approach is less restrictive to allow inclusion of success factors intrinsic to the individual. These are sometimes referred to informally as the "soft skills." In focus groups, these are the characteristics most typically identified by subject matter experts (SMEs) as being the ultimate competencies that great workers demonstrate.

The Filtered employed a four-factor framework of job relevant behavior that has been shown to generalize across jobs. (Saville, 2005) These include:

A) Thought
B) Influence
C) Adaptability
D) Delivery

By reviewing these broad categories, the Filtered insures that the domain of critical success factors for software engineers was included in the assessment framework.

General competencies identified from job analyses and established frameworks are more difficult to assess objectively in comparison to knowledge or specialized skills. They are inherently influenced by context and opinion. What works in one setting, doesn't in another and where judgement drives evaluation, human error (statistical) intervenes. Nevertheless, they are causal to success on the job for programmers and can't be overlooked simply because they aren't easy to agree on. In some cases, general competencies become the differentiating factor if the variance between candidate scores on the technical aspects of the assessment is minimal. Lastly, colleagues with different technical skill sets (i.e., not coding) evaluate what they know. These are important no matter how technically skilled one may be.

The Filtered Approach to Assessing General Competencies.

The Virtual Interview. Following the performance-based "test" of technical skills, the Filtered provides the option of presenting a number of questions to candidates via prerecorded audio. The presentation of questions is followed by the capture of the candidate's response via audio

[6] Onetonline.org Software Developers. (2017)

recording. This, question and answer, exchange forms a "virtual interview," capturing data both complementary and supplementary to the programming assessment.

Policy Capturing. Complementary items follow the practice of policy capturing. This allows the candidate to describe their approach, alternative considerations and any other information they would like reviewers to know that couldn't be captured by the work product alone. Reviewers benefit from gaining a better understanding of the candidate's thought process while solving tasks – especially when the tasks include multiple steps. Sequential contingencies between steps used to solve the problem can be isolated and factored into the candidate's solution beyond the actual code and solution generated (or not).

Behavioral Event Interview. The inclusion of supplementary items follows the technique of behavioral event interviewing (BEI). These interviews explicitly instruct and/or lead candidates to respond to interviewer questions or requests with specific examples (events) illustrative of the competency being assessed. Speculation, opinion, indirect reference, etc., are explicitly discouraged. This technique of posing questions has been shown to improve the reliability and validity of open-format items – even amongst highly trained interviewers. This is the point where the non-technical aspects of the job are assessed. As with the coding work sample, content for use in the competency interview is available to select from an extensive library of items built according to specifications of posted job descriptions pertaining to programmers as well as formal job analyses. (previously described)

Scoring. Whereas the coding test is objective and best scored by Filtered Ai's algorithms, clients represent the most appropriate source for evaluating candidate responses to interview questions for two reasons:

1. Interview content is chosen by clients, and frequently supplemented with their own material
2. The Filtered library of interview items reflect best practice in item writing, but differences in client needs should be the ultimate reference for evaluation of candidate responses

Since no score is provided on the interview, the final integration of test performance (coding plus interview) is left to the client. This practice combines the assessment rigor of the Filtered without presumption of client needs, market, etc.

STANDARDIZATION

Standardization is at the core of the Filtered's design. Given a content-referenced approach to validity, standardization is paramount. As referenced earlier, standardization for the fair use of the Filtered cannot be achieved by merely addressing the issue of cheating posed by remote administration. Cheating is addressed, but the Filtered goes well beyond typical un-proctored assessments.

Four principles underpin the Filtered's approach to insuring standardized assessment:

1. Minimize confirmation bias among recruiters
2. Maximize completion rates among candidates
3. Provide a fair, standardized environment
4. Minimize cheating

Confirmation Bias. Resumes and CVs are the addiction of hiring parties. Although Filtered Ai doesn't argue the point that past behavior is the best predictor of future behavior, it is argued that resumes, CVs, and silver-tongued interviewees can actually lead to poor hiring decisions. This conceptual issue has been addressed earlier but Filtered Ai has taken several specific steps to minimize this bias.

1. "Show me" testing. Simulations are the ultimate verification of claimed competence of specialized skills.
2. Reserving candidate information until AFTER they have completed the entire assessment. Without knowledge of the factors that are highly susceptible to assumption, assessment is restricted to the test and interview.

Cheating. Each candidate is filmed from the very beginning of the Filtered to its submission. Remote logging, referencing information beyond the terms of test completion, multiple respondents, and a host of potential means to yield inaccurate scores are minimized by full filming.

Completion Rates. A systematic pattern pertaining to the completion of an assessment can pose a risk to standardization, and therefore fair use.

Most frequently, dropouts are due to performance concerns (which typically are correct), or a lack of motivation (also, valid information). If, however, candidates are not represented by testing in relation to their representation in the market, systematic bias can occur. A primary means of such a case is the ability of candidates to "perform." They must have equal access. And they must have equal treatment.

Given that Filtered can be administered at any time, in any place – and there is a VERY high likelihood that programmers have access to the required technology, the issue of equal access is virtually eliminated. Presentation of the assessment at the complete convenience of the candidate is also a factor likely to raise completion rates.

Candidate Feedback. Although not a formal means of demonstrating fair treatment, feedback from candidates can provide relevant insight.

One form of feedback is built into the assessment – interview questions permitting the candidate to comment or make other statements of their volition regarding the experience.

Here, candidates can, or may, mention a factor that could legitimately pose a “handicap.” It isn’t unusual for false statements to be made in these circumstances, but the Filtered is filmed – from beginning to end. Most of the typical claims of distraction, etc. are either eliminated (candidate aware of filming) or verifiable (candidate lies anyway).

Finally, no reports of mistreatment or procedural inequity have been substantiated with the Filtered. To the contrary, candidates more frequently report being highly satisfied with the instrument since it so clearly represents what they believe they are applying for (working with computer code). They also appreciate the convenience afforded by the Filtered in terms of its completion.

VALIDITY

The Filtered Ai platform assesses both the coding capability and a range of other competencies highly predictive of success as a programmer.

Programming. Computer coding is an exacting task requiring very specialized, learned knowledge and skill. According to arguably the most comprehensive source of job analytic information, O*Net, the core tasks of a growing number of jobs pertaining to the design, development, compilation, testing, review, and administration of computer-based technology are all heavily dependent on these very discrete skill sets. Making computers perform increasingly sophisticated tasks, even to include the feature of self-generating code, or "Artificial Intelligence" so clearly depends on technical skills that even the end-user would agree are unique to programmers. In short, jobs falling into the "computer programming" job-family not only require highly specialized skills, technical skills and knowledge constitute the majority of their core tasks. They are specific, important and pervasive.

Jobs for which core tasks and requisite capabilities draw heavily on declarative or procedural knowledge are well suited for a content-related validation. According to the most definitive source for test validation,[7]

> Evidence for validity based on content typically consists of a demonstration of a strong linkage between the content of the selection procedure and important work behaviors, activities, worker requirements, or outcomes on the job. (pg. 21)

As such, the Filtered is highly suited to a content-referenced validity strategy. Therefore, the validity of the Filtered assessment is a function of the quality of its design.

Applicant Reception. Candidate reactions to selection assessments are important not only to the pragmatism of an assessment, they influence perceptions of procedural justice, or, the fair treatment of applicants. Assessments with content that more closely resembles applicant expectations based on their understanding of the job are viewed much more favorably and fairly than obtuse content with no obvious link to the target job. Challenges to the fairness of these assessments are very rare.

Tests are assessments with right or wrong answers. A true test has only right or wrong answers – there is no judgment involved in the scoring at all. The coding component of the Filtered uses algorithms to determine item-level and total-test scores. There is very little room for perceived deception with formal tests because there is an agreed answer and anyone that provides it, gets the same credit.

[7] *Principles for the Validation and Use of Personnel Selection Procedures*. (Fourth edition, 2003)

Content Validity and the Filtered. Coding language is highly precise. There is no slang, dialect or colloquialism. Only commentary, irrelevant to program compilation, is free to vary. Coding language is more formal than spoken language.

To the extent that the content of any selection test directly represents and is required for acceptable performance on the job in consideration, no better form of validity can be produced than content-related validity. This is because criterion-related correlations can be subjected to factors not included, or adequately controlled, in the true identification of correspondence between test and job. Unspecified error variance can skew criterion-related correlations making them appear more compelling than accurate.

CONCLUSION

Based on the described design, development and recommended application, The Filtered is considered to be appropriate and highly defensible for use in the selection for jobs requiring knowledge and use of computer coding.

R. Chris Steilberg, PhD

CEO, Talentlift LLC