Impact Factor = 4.153, http://www.epitomejournals.com Vol 4, Issue 4, April 2018, ISSN : 2395-6968

Epitome : International Journal of Multidisciplinary Research
ISSN : 2395-6968

A BROAD TAXONOMY OF CURRENT TECHNIQUES TO DISTRIBUTED MONITORING SYSTEMS

MS. SONALILIMBAJI VIDHATE
Research Scholar
Department of Computer
K. K. Wagh College of food technology ,
Saraswatinagar, Panchavati, Nashik - 422 003 Maharashtra,
sonali.vidhate878@gmail.com

ABSTRACT

Distributed and segment based designs are ending up increasingly pervasive PC frameworks. The expanded complexities presented by the circulation hamper reliability, stressing the requirement for confirmation systems custom-made for an appropriated setting. Runtime check has demonstrated to be a feasible methodology for confirming correctness, by concentrating on the adherence of the runtime-produced follow to the ideal properties. We present a wide scientific classification of current strategies to appropriated observing, a broad taxonomy of current techniques to distributed monitoring, coming full circle in the proposition of a novel relocating screen approach. We contend for specific circumstances where this methodology introduces clear points of interest over current procedures.

KEYWORDS

System Features, Dispersed Monitoring, Inert Orchestration, Inert Choreography, Active Orchestration, Active Choreography

RESEARCH PAPER

1. INTRODUCTION

As frameworks become increasingly mind boggling, monolithic structures are ending up less normal and conveyed and part based frameworks are winding up more standard. Dispersed models present extra complexities, for example, calculation/memory circulation, worries for data classification and engineering dynamicity. Such issues hamper framework constancy and heartiness, underscoring the requirement for procedures ensuring accuracy customized for conveyed frameworks. Programming confirmation procedures generally incorporate testing, model checking and runtime check. Despite the fact that testing is halfway alluring because of its versatility, it is inadequate because of (I) its absence of program inclusion, whereby testing can just discover the nearness and not demonstrate the nonappearance of bugs [8], (ii) the trouble of viable experiment age, which when joined, offer 'sensible' inclusion of conceivable framework conduct. Model checking gives the most noteworthy assurances yet the state space blast required by the displaying of even modestly measured frameworks makes this methodology unreasonable as a rule. This issue is additionally exacerbated by the asynchrony and simultaneousness innate in appropriated structures [13].

Runtime substantiation [4, 11] is troubledwith officially confirming the framework follow generated at runtime. This procedure is brought out through an executable screen confirming the generated follow against a lot of alluring properties, with the framework ensured to never go past an awful state undetected. Preferences with runtime confirmation incorporate (I)the way that it guarantees that the system might be halted the minute issues are distinguished in a tractable way, (ii) follow age is left to the framework, (iii) check proceeds past framework organization.

The resulting paper explores the utilization of runtime check to a disseminated setting, while at the same time proposing a novel relocating screen approach we accept is invaluable for observing certain situations of appropriated frameworks. section 2 presents dispersed framework attributes appropriate to the plan of a forthcoming checking structure, while additionally presenting an inspiring model. These go about as our reason for looking at the different observing methodologies in Section 3, where we likewise layout circumstances where each approach is most appropriate. At long last, area 4 closes the paper with bearings for future work.

2. SYSTEM FEATURES

We think about circulated frameworks with a lot of independent, simultaneously executing sub-frameworks imparting through message passing. Each sub-framework has (I) its very own execution string, (ii) neighborhood memory, and (iii) classified nearby data. Most web based and administration arranged frameworks promptly fit in the above engineering, as do frameworks holding fast to the Enterprise Service Bus structures [6]. Attributes relevant to this type of engineering influence the plan of a forthcoming checking system, and are talked about underneath:

Design and memory dispersion: Distributed system structures involve calculation that is circulated over its different computational elements, just as the parceling of the framework's worldwide state among a lot of remote segments. Essentially, the framework's worldwide state isn't promptly accessible and worldwide state projection is regularly unreasonable, because of the voluminous data move included and the limitations on the correspondence medium.

The correspondence medium: Communication between physically dispersed subsystems is significantly slower than nearby correspondence, constrained by limited data transmission confinements. Thus, an effective disseminated framework should concentrate on limiting remote correspondence. In addition, some correspondence media may not save message request during correspondence. Different qualities that one ought to consider incorporate correspondence synchrony just as the potential for non-lossy correspondence.

Data region: Distributed systems might be made out of subsystems each containing classified nearby data. This is particularly valid in heterogeneous conditions, where disputes of trust are common. It is the obligation of any imminent checking system to regard data area, since inability to do prompt so extra information introduction. Information presentation can appear as (I) introduction through remote correspondence crosswise over perilous mediums, (ii) introduction of secret data between non-special subsystems.

System Physiography: This identifies with the framework's design dynamicity i.e., regardless of whether the framework concedes an engineering which 'develops' during execution. Framework design may develop in one of two different ways; (I) the quantity of contributing computational elements changes during execution, (ii) the correspondence design between sub-system changes. Systems which concede dynamic setups incorporate shared frameworks, just as administration arranged structures utilizing intermediaries for administration query.

An Inspiring Citation

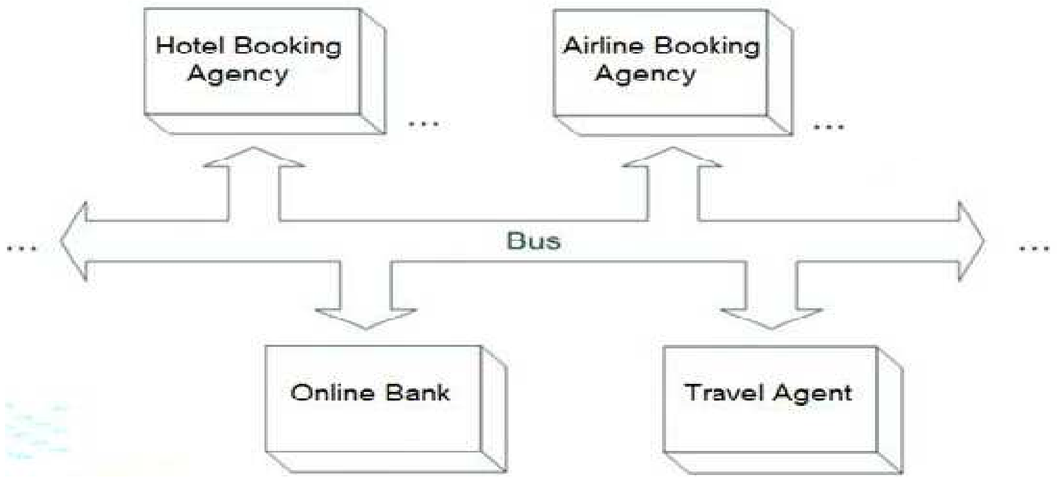

Figure 1: The Travel Agent.

Figure 1 delineates a common conveyed system whereby a trip specialist is in charge of booking occasions for the customer's benefit. Given a lot of customer demands and money related impediments, the operator's undertaking is to scan for arrangements over various lodging and carrier booking offices, booking the best bargain (through the customer's online bank) given the predefined limitations.

We will take a glimpse at two variations of the above situation. In the principal situation, the trip specialist is to speak with a pre-decided arrangement of online banks, just as inn and carrier booking organizations. The subsequent situation includes the specialist going about as an agent, progressively looking for inn and aircraft booking organizations as per the customer's solicitations, just as speaking with the suitable bank in charge of the customer's record. Unmistakably, the subsequent situation is increasingly adaptable and could conceivably return better outcomes. Nonetheless, extra abilities come at the expense of intricacy; though all contributing elements in the primary situation are known from the earlier, elements engaged with the subsequent situation must be found at runtime. We will later perceive how the two situations influence the pertinence of appropriated checking approaches.

The convention clung to by the trip agent while providing food for a customer solicitation is as per the following:

1. The customer gives the trip specialist (I) financial balance subtleties (for future exchanges), and (ii) a lot of parameters in regards to the ideal occasion (goal nation, cost and so on).
2. In light of this data, the trip specialist associates with (I) the customer's online bank, (ii) up-and-comer flight and lodging booking offices.
3. Offices are picked dependent on some ideal choice arrangement (conceivably affected powerfully dependent on some administration query), and questioned for a citation of wanted appointments.
4. with the citations close by, the trip specialist in this way questions the bank if the customer's financial balance can bear the cost of the gave bundle. Provided that this is true, the trip specialist comes back to the customer for affirmation.
5. Whenever affirmed, exchanges are activated for the customer's sake; else the procedure restarts by picking distinctive flight and carrier booking offices.

The trip agent model shows the qualities talked about above. Obviously, the system is dispersed, since both the computational elements and the framework's memory space is apportioned into a lot of disseminated sub-systems. In addition, the thought of data classification is of extensive significance. Both the online bank, also reserving offices concede nearby data whose protection of area is principal. Information presentation can take the two structures for example introduction of secret data, for example, financial balance data crosswise over temperamental mediums for this situation; the web, just as presentation crosswise over elements for instance, rival booking organizations. Given that between framework correspondence happens on the web, this suggests the worldwide framework works inside confined transfer speed impediments. At long last, while the principal trip specialist situation concedes a static topology all substances are known preceding calculation, the subsequent situation including administration query concedes a unique topology, since taking part elements rely upon the customer's solicitations and subtleties. Framework accuracy in the model above is of basic significance. One property expected to hold is that of advancement i.e., every customer solicitation submitted to the trip specialist is in the end countered by an idea from inn and flight booking organizations inside a specific time period. Another rightness property includes guaranteeing that the expense of proposed appointments doesn't surpass the customer's bank balance. Albeit the two properties indicate a type of occasion successively, they change on one inconspicuous point; though the previous alludes to by and large non-secret data booking offers are generally freely accessible on the web), the last requires the treatment of private data.

3. DISPERSED MONITORING

Checking accuracy the two properties laid out in Section 2.1 is vigorously impacted by both the hidden appropriated engineering just as the property's temperament. Calculation/memory appropriation powers observing be done crosswise over hazardous mediums. In addition, the checking structure never again has prepared access to the framework's worldwide state, implying that assessing properties over the apportioned state is testing. The correspondence medium may likewise possibly present new issues. It is the duty of all imminent observing structures to limit transmission capacity overhead actuated by the structure inability to do so could meddle with the framework's reconciliation exertion, conceivably adjusting the framework's conduct. In addition, absence of correspondence request safeguarding could prompt screens remotely watching framework conduct into inaccurately approving broken properties, or the other way around (see [13]).

The observing structure is additionally in charge of safeguarding data territory within the sight of classified neighborhood data. At last, the issue of framework setup dynamicity exhibits an extra multifaceted nature, since this requires the observing structure to 'keep up' with the frequently eccentric runtime changes the framework experiences during execution. Directly, there exist various designs and apparatuses for conveyed runtime verification and monitoring. These are best comprehended and sorted by the accompanying two criteria.

Step design vs orchestration: Current ways to deal with dispersed observing can be extensively delegated organization or movement based. In coordination based methodologies, check obligation lies immovably with a focal screen catching all data relevant to the framework's worldwide rightness, as found in figure 2. In spite of the fact that this methodology works flawlessly on solid frameworks, its application isn't as direct in an appropriated domain. For the situation where the checked property concerns just open correspondence between subsystems, this methodology functions admirably by building a screen catching all such correspondence, altering its state as needs be. In any case, when the framework property includes nearby subsystem data, this methodology is not exactly perfect.

To begin with, communication of nearby private data crosswise over remote areas prompts data disclosure. Besides, the volume of data required for incorporated observing is significant, regularly bringing about irrational transfer speed overhead. At long last, coordinated methodologies represent a security hazard by introducing an essential issue of assault, as the screen, through which delicate data can be tapped.

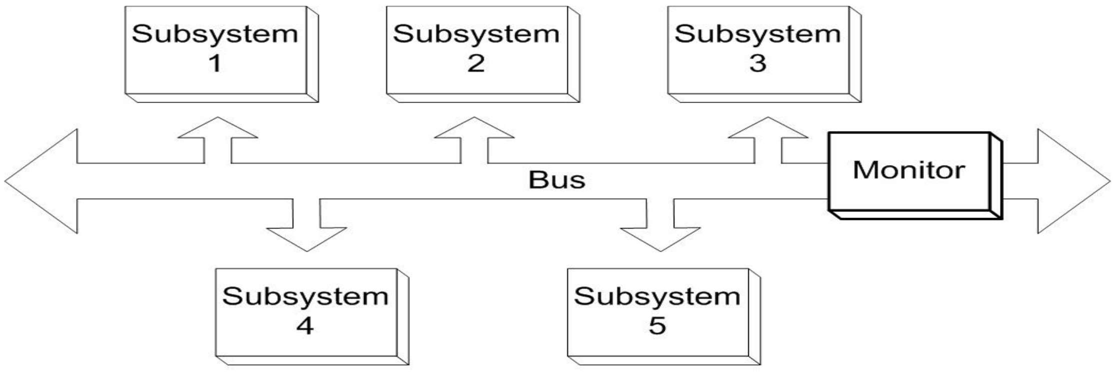

Figure 2: An arrangement based methodology.

Step design based monitoring adopts a more dataflow subordinate strategy, whereby (sub)system occasions drive the execution control stream of the monitoring procedure, regularly prompting a dissemination of observing usefulness over the circulated framework. As a rule, movement based observing can relieve weaknesses presented by arranged methodologies. A movement based methodology can be streamlined to push check to happen locally, limiting information presentation. Likewise, observing limitation kills the necessity of subsystem data move back to a focal screen. This doesn't prevent restricted screens from conveying over the correspondence medium, anyway the volume of data for screen synchronization is typically generously not as much as that required by a focal screen, suggesting that a movement based methodology possibly diminished transmission capacity overhead in specific circumstances. At long last, evacuating the focal screen destroys the security danger of giving a focal assault point. By the by, in spite of the fact that movement based methodologies gloat preferences over coordinated partners, applying movement is frequently increasingly complex particularly since the hubs wherein checking is to be set up must in some way or another empower the instrumentation of observing code, and ought to consequently be utilized distinctly in situations where it is worthwhile to do as such.

Static versus dynamic properties: Static properties incorporate a property whose determination is altogether known at gather time, and stays unaltered during framework execution. Then again, dynamic properties include framework properties (I) whose portrayal are not so much known (or changes) at runtime, or (ii) might be adapted altogether during execution — hence making the screen parameterized by properties which may just accessible at runtime. One discovers dynamic properties, for example, in security-related interruption recognition situations [7], where suspicious client conduct must be scholarly at runtime in the wake of watching the framework to realize what average conduct resembles. Dynamic properties can likewise be logical for example properties which develop as per gathered data.

Taking the travel specialist situation, it is possible that various banks would require the confirmation of various security arrangements, or that the security strategy to be checked relies upon the measure of cash engaged with the exchange (thus, a unique property which relies upon its specific circumstance — the bank and the measure of cash included). In such cases, the property can be viewed as either a complex contingent static property or various more straightforward properties, just one of which is activated progressively at runtime. In spite of the fact that the issue of static versus dynamic properties will undoubtedly appropriate checking structures, dynamic properties have a specific partiality to circulated frameworks because of the probability of dynamic designs — conveyed frameworks whose arrangement advances during execution may require properties which change as needs be in order to screen the powerfully evolving engineering. Plainly, albeit dynamic properties are more expressive than their static partners, they likewise speak to a class of impressively increasingly complex properties to screen, and should possibly be viewed as when vital.

These conditions lead to the plausibility of four classes for distributed monitoring, in particular static organization, static movement, dynamic arrangement and dynamic movement. The decision of methodology regularly relies upon need, contingent upon both hidden framework attributes and the property under concern.

3.1 Inert Orchestration

Considerately the least difficult methodology, static coordination includes utilizing a focal screen catching data over the correspondence medium, and checking a lot of pre-decided properties. This methodology is confirmed in [3], where web administration arrangements actualized in BPEL [2] are checked in coordinated design. Points of interest with this methodology incorporate (I) its oversimplified nature, both in idea and as a rule in application, and (ii) its pertinence when checking properties managing open data over the correspondence medium. In any case, static arrangement concedes predominant issues examined previously. To be specific, static arrangement may prompt information presentation, represents a security chance, could likewise conceivably bring about irrational data transfer capacity overhead and is additionally unequipped for dealing with dynamic properties (henceforth, no powerful designs).

Nevertheless, an inert orchestration based methodology is relevant in specific situations. One could, for instance, screen the trip specialist (expecting the primary situation including a static topology) for the advancement property, by introducing a focal screen which catches open customer demands and offers made by the booking organizations, and confirming that each solicitation is met with a reaction. Notwithstanding, checking the second trip specialist situation (conceding a powerful topology) is unattainable, since static property particulars are unequipped for communicating properties over unique designs.

3.2 Inert Choreography

Inert Choreography includes changing over system properties at gather time into a lot of disseminated screens enveloping the worldwide observing system, as found in figure 3. These screens watch framework conduct locally, and synchronize remotely to accomplish worldwide check of the framework. Current static movement based methodologies incorporate [13, 12, 9, 10, 14].

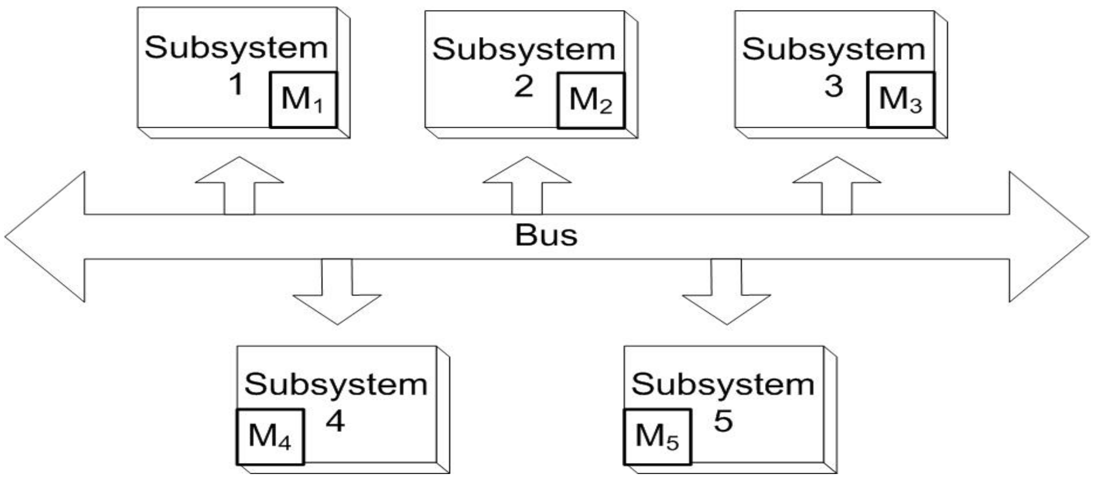

Figure 3: A static choreography-based approach.

Points of interest with static choreography incorporate those talked about for movement by and large, for example (I) safeguarding of region, (ii) conceivable data transmission overhead decrease, and (iii) the expulsion security dangers identified with focal assault focuses. On the off chance that we needed to screen the principal trip specialist situation for the property expressing that appointments don't surpass the customer's monetary restrictions, one could utilize neighborhood screens (one at each reserving office and online bank), with screens at the booking offices telling the screen situated at the customer's specific bank of proposed appointments, which would then be able to check locally that cost doesn't surpass impediments.

Notice how utilizing this methodology no classified customer data leaves the bank's area, rather than a static coordinated methodology which would require move of customer data remotely to the focal screen. The volume of data move for checking objects is additionally diminished, since screen synchronization after booking age includes less data than the exchange of significant bank, inn and flight booking data to a focal screen. Given that screen circulation happens once from the earlier to framework execution, a static movement approach is unequipped for taking care of developing framework properties, or properties learnt at runtime. Taking care of dynamic properties through a static movement based methodology would require (I) Recompilation, and (ii) re-dispersion of the checking structure upon each update the observed properties, which is commonly unfeasible. An immediate ramifications of this announcement is that static movement is unequipped for dealing with dynamic topologies, since one would require dynamic properties fit for evaluating over advancing framework structures. This infers observing the second trip specialist situation is unattainable utilizing static choreography.

To conclude, note how the treatment of the advancement property talked about for static coordination is likewise feasible utilizing static movement, by coordinating a customer demand at the trip specialist with booking reactions at inn and flight booking organizations. Be that as it may, no clear preferred position is picked up by applying static movement over static arrangement in this situation, making the use of movement a pointless difficulty.

3.3 Active Orchestration

Energetic Orchestration based methodologies include the reception of a focal screen remotely watching sub-framework conduct, which anyway takes into account the checking of dynamic properties. An occasion of dynamic arrangement is seen in [1], including the concentrated checking of web benefits through the particular of BPMN work processes [5]. Besides, this methodology takes into consideration the organization of the confirmation of agreements (speaking to framework properties) on-the-fly, taking into account the check of dynamic properties. The principle preferred position of dynamic coordination over its static partner is the capacity to deal with dynamic properties. A unique coordinated methodology may for instance screen the second trip specialist situation for both recently talked about properties, with the focal screen tuning in to data from new offices found by the trip specialist appropriately. By and by, dynamic arrangement still experiences information presentation, is possibly wasteful because of irrational transfer speed overhead, and still speaks to a security chance by showing a special element, as the screen, through which data can be tapped. Albeit conceivable, a powerful arranged methodology for checking the second trip specialist situation is subsequently inadmissible, since utilizing such a methodology would in any case require remote exchange of delicate customer bank subtleties to the focal screen. A unique coordinated methodology seems to fit best when just open data should be checked over the correspondence medium some powerful properties.

3.4 Active Choreography

Correspondingly to static choreography, dynamic movement involves the circulation of observing usefulness over the dispersed framework. Be that as it may, with dynamic movement appropriation happens during execution, subsequently taking into account the re-conveyance of the observing system taking into consideration the checking of dynamic properties. As far as we could possibly know, directly no current observing design falls in this class.

With this impact, we propose the investigation of dynamic movement using relocating screens for example screens running locally to sub-frameworks, and physically relocating to different areas when requiring checking neighborhood conduct relevant to the worldwide rightness of the framework, as delineated in Figure 4.

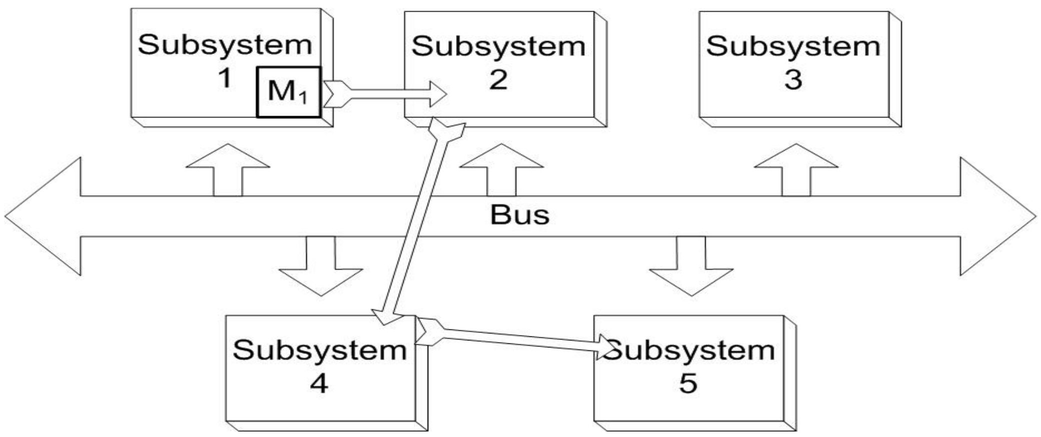

Figure 4: A moving screen approach.

Migrating screens adopt a property skeptic strategy, with the end goal that each sub-framework is instrumented to uncover a letters in order of data accessible to the screens solely at a neighborhood level. Screens are consequently permitted to relocate to sub-framework areas, locally perusing data relevant to the property under thought. This methodology takes into account the checking of properties learnt at runtime, since new properties can be changed over to comparing moving screens, and are in this way executed on-the fly without the requirement for framework restart. Dynamic movement is accomplished through a thoughtfully basic administrator activating screen relocation. This administrator takes into account the runtime system redistribution, since screens can be indicated to relocate to interchange areas during execution, perhaps even areas not known toward the beginning of calculation. Relocating screens are thus fit for taking care of dynamic designs, by moving to new areas once coordinated with the worldwide engineering. This, thusly, is a shortcoming of dynamic movement approach, since the hubs should be eager to introduce screens known uniquely at runtime. This prompts an issue of trust, since the screen may begin from a deceitful gathering — then again, one can expect screens to be marked by reliable accomplices, or even apply static investigation systems or require the screen to show up as evidence conveying code to make the methodology viable.

A moving screen method is profitable in three regards 1. it concedes preferences relating to movement based methodologies — see Section 3.2. Being movement based methodology, moving screens take into consideration the conservation of area while additionally conceivably decreasing data transfer capacity overhead in specific circumstances.

2. It builds adaptability, i.e., the capacity of the checking structure to adjust to unforeseeable (at assemble time) changes during framework execution, at minimal extra multifaceted nature. This is accomplished by including the relocation crude. As talked about, changes during framework execution can appear as unique framework setups, too powerful properties (counting properties learnt at runtime and properties which advance during execution).

3. It accomplishes raised epitome. Relocating screens work at a more elevated amount of reflection, accomplishing movement while restricting together computational substances and data (screen express) whose design is that of accomplishing one shared objective (checking of a property). Reasonably, this is a sharp qualification from static movement based methodologies, since screen calculation and state working to confirm a specific property are frequently circulated all through the framework. Note this doesn't stop the moving screen come nearer from utilizing different simultaneous screens. Raised embodiment additionally indicates moving screens being conceivably progressively agreeable to adaptation to internal failure systems, since relocating screens are instinctively reconfigurable on-the-fly. This is an alluring property within the sight of fractional disappointment innate in circulated frameworks. Consequently, one could for instance modify a screen's transitory examples at runtime once a contributing framework element is regarded to be inaccessible. Another result of raised exemplification is conceivably simpler upkeep (instead of keeping up a static movement based methodology), since refreshing a structure utilizing moving screens would just require update of the relocating screens, as a rule far less in number than the measure of updates included when refreshing a static movement based system (one screen is allocated to every element). Give us a chance to consider the checking of the second trip specialist situation for the second property that the expense of appointments doesn't surpass the customer's bank balance, utilizing a moving screen approach. One could characterize a moving screen, whose state involves (I) the customer's online bank

data, (ii) inn and flight booking organization data under thought, (iii) a counter recording the expense of proposed appointments. Execution of the moving screen would begin at the trip specialist area, gathering data with respect to contributing elements during administration of the customer's solicitation (bank, booking organizations). Utilizing this data, the moving screen can then powerfully set area data where the screen is to move to, beginning with relocating to contributing booking organizations. At each reserving office, the screen gathers the expense of the specific organization's recommended booking (cost is aggregately put away utilizing the screen's counter). When all reserving expenses are gathered, the screen at last relocates to the online bank, and utilizing the customer's online bank data checks whether the customer can bear the cost of the recommended appointments. Note how data area is saved while most likely bringing about transfer speed overhead decreases (instead of arranged methodologies), since all inclusive observing the property has been diminished to nearby checking sprinkled with a couple of screen movements.

Confirmation of dynamic properties has additionally been trivialized using the movement administrator related to a property rationalist methodology. New reserving organizations (found by the trip specialist through administration query) are promptly observed by taking into consideration the screen to relocate at the new areas.

A migrating monitor approach additionally concedes inconveniences which settle on its application an imperfect decision in specific circumstances. Right off the bat, the methodology is most appropriate when the property under thought concedes a moving screen portrayal dependent on significant neighborhood checking and couple of remote relocations. On the other hand, properties requiring moving screens dependent on an outlandish measure of movement may bring about considerable transmission capacity overhead. The relocating screen approach is additionally founded on the inalienable suspicion that various subsystems trust code substances to move and run locally, while approaching touchy neighborhood data. Unmistakably, this may not generally be the situation, since framework directors may object to outside executable substances having favored access to their frameworks. At long last, in spite of the fact that the property skeptic approach is adroitly favorable for reasons talked about above, it is a more perplexing instrumentation approach than that applied in progressively conventional runtime confirmation methods, (for example, in [11, 4, 13]), which could bring about higher likelihood of disappointment.

For example, in spite of the fact that confirmation of both trip specialist situations for adherence to the principal property is additionally conceivable utilizing moving screen approach, it isn't beneficial to do as such. Given that the primary trip specialist situation concedes a static topology it is inefficient to apply a methodology concentrated on dealing with dynamic properties, inferring that a methodology taking care of static properties would get the job done. In addition, the primary property includes open data moved over the correspondence medium, suggesting that data area is a non-issue. Thus, a coordinated methodology would do the trick.

Taking everything into account, migrating monitors offer a substitute way to deal with appropriated checking, whose materialness is best when confronting severe data secrecy limitations in an exceptionally powerful condition (either regarding the property being confirmed, or the fundamental framework conceding dynamic setups).

4. CONCLUSIONS AND FUTURE WORK

Runtime monitoring of distributed systems is impressively increasingly unpredictable instead of solid neighborhood framework. In this paper, we show how this is intensely affected by attributes natural for circulated frameworks. We additionally propose what we accept to be a novel relocating screen approach, upholding the utilization of area mindful screens tuning in to occasions solely at a neighborhood level, before moving to different areas when their conduct winds up appropriate to the framework's general accuracy. We contend that this methodology regards data region and handles dynamic properties.

We are presently examining conventional properties of this procedure, for example, (I) checking doesn't influence calculation, (ii) neighborhood observing jelly area, (ii) nearby checking is equal to worldwide observing, disregarding area data. Confirmation of these three system properties should fill in as once-overs to verify everything is ok to discover appropriateness of the methodology. We are likewise investigating the augmentation of Larva [11] to deal with relocating screens to be applied to contextual analyses utilizing Enterprise Service Bus middleware [6].

REFERENCES

Charlie Abela, Aaron Calafato, and Gordon J. Pace.Extending wise with contract management.In WICT 2010.

Alexandre Alves, Assaf Arkin, Sid Askary, Ben Bloch, Francisco Curbera, Yaron Goland, Neelakantan Kartha, Sterling, Dieter K¨onig, Vinkesh Mehta, Satish Thatte, Danny van der Rijn, Prasad Yendluri, and Alex Yiu. Web services business process execution language version 2.0. OASIS Committee Draft, May 2006.

Fabio Barbon, Paolo Traverso, Marco Pistore, and Michele Trainotti. Run-time monitoring of instances and classes of web service compositions. In ICWS '06: Proceedings of the IEEE International Conference on Web Services, pages 63–71, Washington, DC, USA, 2006. IEEE Computer Society.

Howard Barringer, Allen Goldberg, Klaus Havelund, and Koushik Sen. Rule-based runtime verification.

pages 44–57. Springer, 2004.

Marco Brambilla, Stefano Ceri, Piero Fraternali, and Ioana Manolescu. Process modeling in web

applications. ACM Transactions on Software Engineering and Methodology (TOSEM), 15:360–409, 2006.

David Chappell. Enterprise Service Bus. O'Reilly Media, June 2004.

Dorothy E. Denning. An intrusion-detection model. IEEE Transactions on Software Engineering, 13:222–232, 1987.

Edsger W. Dijkstra. Notes on Structured Programming. April 1970.

Ingolf H. Kruger, Michael Meisinger, and ¨ Massimiliano Menarini. Interaction-based runtime verification for systems of systems integration. Computer Science and Engineering Department, University of California, San Diego USA, July 2008.

Masoud Mansouri- Samani and Morris Sloman. Gem: a generalized event monitoring language for distributed systems. Distributed Systems Engineering, 4(2):96–108, 1997.

Gordon Pace, Christian Colombo, and Gerardo Schneider. Dynamic event-based runtime monitoring of real-time and contextual properties.In 13th International Workshop on Formal Methods for Industrial Critical Systems (FMICS'08), LNCS 4916.Springer-Verlag, 2008.

Thomas S. Cook, Doron Drusinksy, and Man-TakShing. Specification, validation and run-time moniroting of soa based system-of systems temporal behaviors. In In System of Systems Engineering (SoSE).IEEE Computer Society, 2007.

Koushik Sen, Abhay Vardhan, Gul Agha, and GrigoreRo¸su. Efficient decentralized monitoring of safety in distributed systems. Software Engineering, International Conference on, 0:418–427, 2004.

Wenchao Zhou, Oleg Sokolsky, Boon Thau Loo, and Insup Lee.Dmac: Distributed monitoring and checking. In Saddek Bensalem and DoronPeled, editors, RV, volume 5779 of Lecture Notes in Computer Science, pages 184–201. Springer, 2009.